Numerical solution of thin-film flow equations using adaptive moving mesh methods

Abdulghani Ragaa Alharbi

Submitted in partial fulfilment of the requirements of the degree of

Doctor of Philosophy

Keele University

School of Computing and Mathematics

March 2016

Acknowledgement

This thesis would not have been possible without the support of my supervisor, Dr. Shailesh Naire. First of all, I would like to express my sincere gratitude to Dr. Shailesh Naire, for the continuous support during my PhD study and research, for his motivation, enthusiasm and immense knowledge. I very much appreciated all the patient guidance and encouragement and advice he has provided throughout my time as his student. In addition, I would like to thank my family, brothers, sister, nephews and nieces, for their patience and encouragement they have provided to me throughout my PhD research. I wished to tell my parents and my brother Abdullah, who died before I have finished my research, that I very much appreciated all their support and encouragement. I specially thank my wife and children, Saja, Alwaleed and Danah, who experienced all of the ups and downs during my research. Last but not least, I would like to say thank you very much to everyone who kept in touch with me from beginning of my study until I finished my research.

Abstract

Thin liquid films are found everywhere in nature. Their flows play a fundamental role in a wide range of applications and processes. They are central to a number of biological, industrial, chemical, geophysical and environmental applications. Thin films driven by external forces are susceptible to instabilities leading to the break-up of the film into fingering-type patterns. These fingering-type patterns are usually undesirable as they lead to imperfections and dry spots. This behaviour has motivated theoreticians to try to understand the behaviour of the flow and the mechanisms by which these instabilities occur. In the physically relevant case when surface tension is large, the film's free surface exhibits internal layers where there is rapid spatial variation in the film's curvature over very short lengthscales and away from these internal layers the film's curvature is almost negligible. This provides the main motivation for this thesis which is to develop adaptive numerical solution techniques for thin film flow equations that fully resolve such internal layers in order to obtain accurate numerical solutions. We consider two thin film flow problems in one and two-dimensions to test the adaptive numerical solution techniques developed in this thesis. The first problem

we consider is related to a liquid sheet or drop spreading down an inclined pre-wetted plane due to influence of gravity. The second problem we consider is also related to the spreading of a liquid sheet or drop down an inclined pre-wetted plane including surfactant-related effects in addition to gravity. We follow the r-adaptive moving mesh technique which uses moving mesh partial differential equations (MMPDEs) to adapt and move the mesh coupled to the underlying PDE(s). We show how this technique can accurately resolve the various one and two-dimensional structures observed in the above test problems as well as reduce the computational effort in comparison to numerical solutions using a uniform mesh.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Thin liquid films are found everywhere in nature. Thin liquid film-flows play a fundamental role in a wide range of applications and processes. They are present in industrial, biological, chemical, geophysical and environmental applications. Some examples are, for instance, in industrial applications such as, coating flows (e.g. drying of paint) [68, 69] and computer microchip production [75], in biological applications such as, the mucus layer lining the airway walls of the lung which assists in conditioning and sensing of air before it accesses the lung [46, 45] and is essential for the stability the lung, in geophysical and environmental applications such as, mudslides and lava flow [2, 44]. It is important to understand the mechanisms controlling the fluid flow within these liquid films through the study of such applications.

When the films are subjected to the action of various mechanical, thermal, chemical or structural forces, they display interesting dynamics such as, wave propagation, wave steepening and, spatial and temporal instabilities. The interested reader is referred to the review articles by Oron et al. [71] and Craster & Matar [26] for a review on these. A particular class of thin film flow problems that has a wide range of applications and is of particular interest both in the physical and mathematical context are those that include surface tension effects [69]. In the physically relevant case when surface tension is large, the film's free surface exhibits internal layers where there is rapid spatial variation in the film's curvature over very short lengthscales and away from these internal layers the film's curvature is almost negligible. This provides the main motivation for this thesis which is to develop adaptive numerical solution techniques for thin film flow equations that fully resolve such internal layers in order to obtain accurate numerical solutions. We consider the following two thin film flow problems to test the adaptive numerical solution techniques developed in this thesis. The first problem we consider is related to a liquid sheet or drop spreading down an inclined pre-wetted plane due to influence of gravity. For example, rain drops or sheet of rain running down the windscreen of cars. The interesting phenomenon in the case, is the breaking up of the sheet of liquid into fingers (see figure 1.1). The important physics related to the spreading fluid drop or sheet down an inclined plane are gravity and surface tension. Figure 1.2(a, b) illustrate the evolution of the cross-section of a sample finger shown in figure 1.1. We observe that the shape of the bulk drop is influenced by gravity with surface tension effects important near the drop's leading and trailing

edges. Near the leading edge the effect of surface tension results in a hump which is referred to as a capillary ridge. There are capillary wave near the leading and trailing edges also due to surface tension (of a very short lengthscale so cannot be seen on the scale shown).

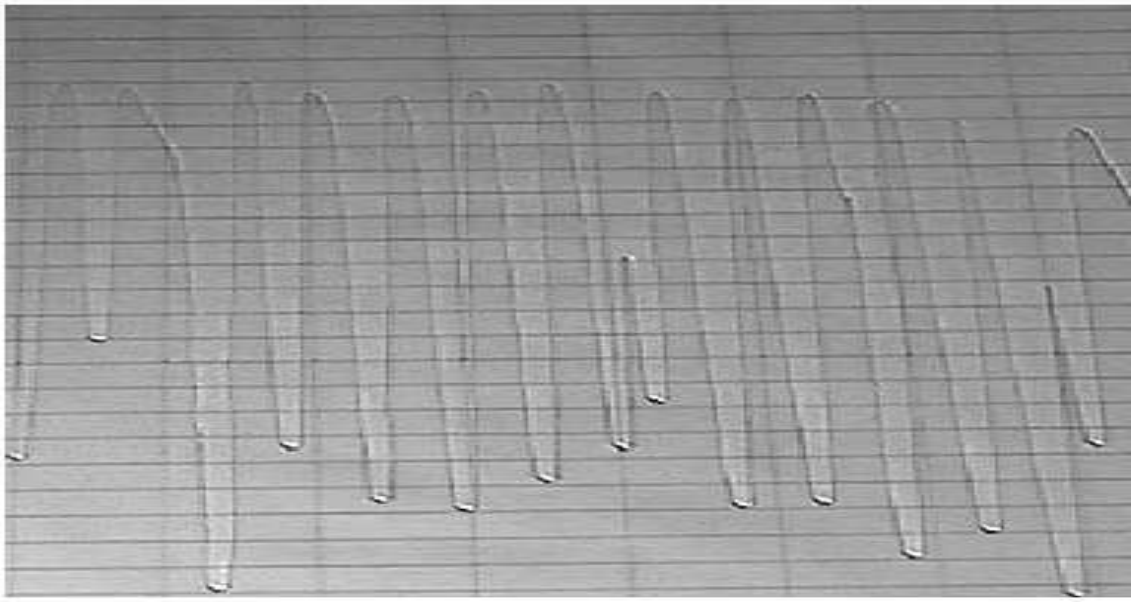

Figure 1.1: A viscous liquid sheet spreading down an inclined pre-wetted plane due to gravity. Notice that the liquid sheet breaks up into fingers. This figure is reproduced with permission from L. Kondic [57].

The second problem we consider is also related to the spreading of a liquid sheet or drop down an inclined pre-wetted plane including surfactant-related effects in addition to gravity. A more dramatic fingering instability now arises when a drop laden with surfactant spreads on a horizontal pre-wetted plane (see figure 1.3 of an experiment by Afsar-Siddiqui *et al.*[1]). Surfactants are chemicals that adsorb into free surfaces and reduce the surface tension there. Gradients in surfactant concentration result in gradients in surface tension which can also drive flow (referred to as Marangoni

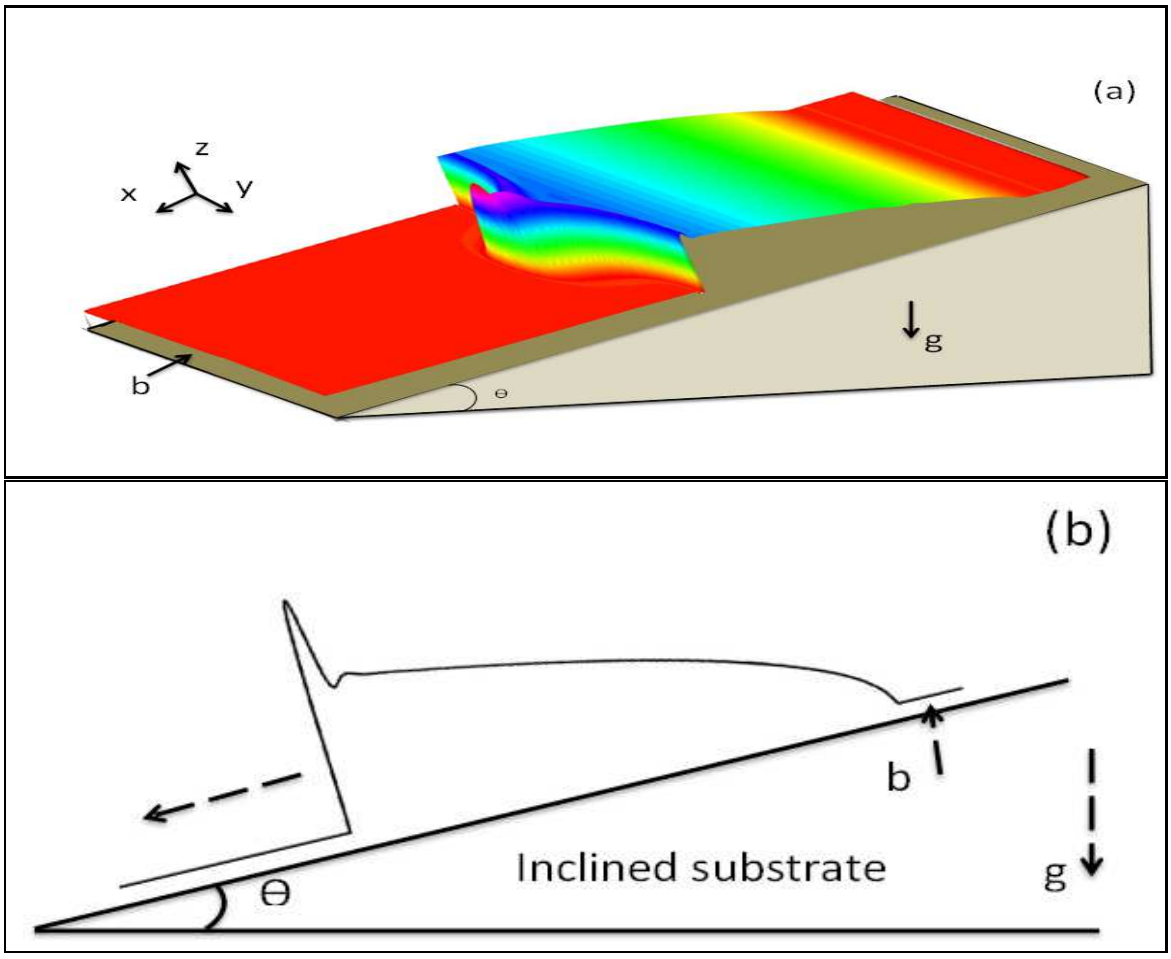

Figure 1.2: This figure illustrates a schematic of the fingers shown in figure 1.1 showing (a) side view and (b) single finger.

flow). This lead to the fingering patterns observed in figure 1.3. Figure 1.4 shows a cross-section of a single finger shown in figure 1.3. The dynamics in this case is much different from that shown in figure 1.2(b). We observe an advancing fluid front ahead of the bulk drop mainly due to surface tension gradient-driven flow. Surface tension effects are only important near the bulk drop's leading edge resulting in a capillary wave there. An ultra thin film develops between the bulk drop and fluid front which undergoes severe thinning. At the trailing edge of the fluid front a rarefaction wave develops.

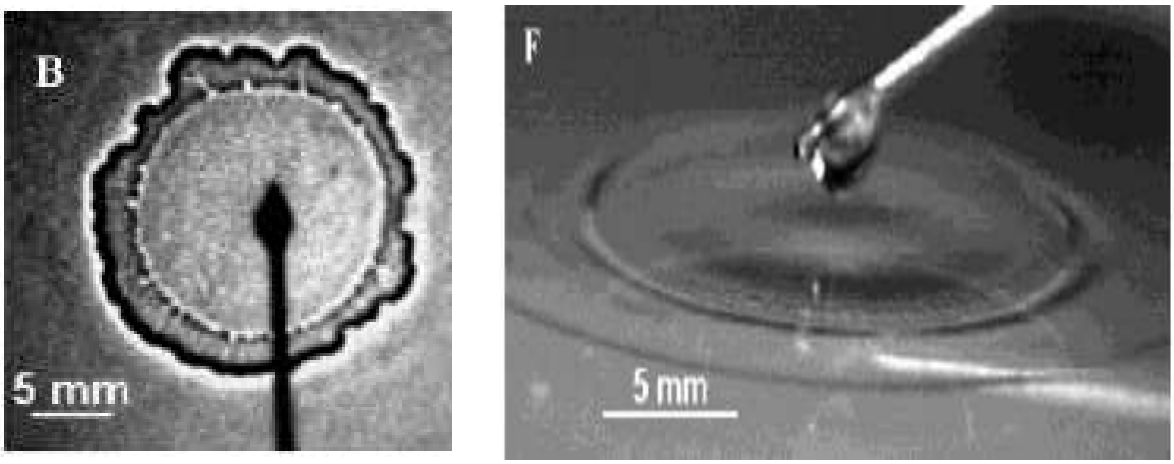

Figure 1.3: This figure illustrates the development of fingering-type patterns in a surfactant-laden drop spreading on horizontal pre-wetted plane. This figure is reproduced with permission from Afsar-Siddiqui *et al.* [1].

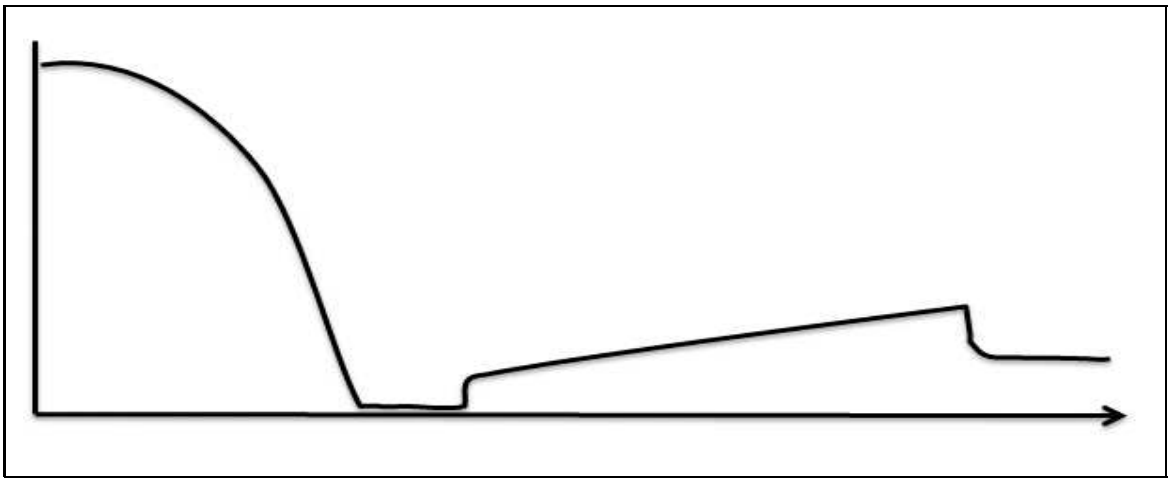

Figure 1.4: This figure illustrates a schematic of the cross-section of one finger shown in figure 1.3.

1.2 Thin film equations for gravity and surfactant-related spreading

The key aspect in thin liquid films is that the characteristic film thickness H (say) is much smaller than the length of film L (say); i.e., the small aspect ratio, $\epsilon = H/L \ll 1$, which is typically much less than one. This is important for using lubrication theory (or long-wavelength analysis) to reduce the Navier-Stokes equations of fluid flow and free surface boundary conditions to an evolution equation for the free surface represented by $h(\boldsymbol{x}, t)$. Here $\boldsymbol{x} = (x, y)$ and t is time. For a liquid sheet or drop spreading down an inclined pre-wetted plane due to gravity, the dimensionless evolution equation can be written to leading order in ϵ as

$$h_t + \boldsymbol{\nabla} \cdot \left[\frac{1}{3} Ca \; h^3 \boldsymbol{\nabla} \nabla^2 h - \frac{1}{3} D(\theta) h^3 \boldsymbol{\nabla} h \right] + \left[\frac{1}{3} h^3 \right]_x = 0. \tag{1.1}$$

For a derivation of this from the Navier-Stokes equation, one can refer to Kondic [57]. We note that Eq. (1.1) is derived using a scaling based on balancing the horizontal component of gravity with viscous forces. This is a fourth order nonlinear degenerate parabolic PDE for the evolution of the film thickness $h(x, y, t)$. It is degenerate since the coefficient of the highest derivative tends to zero as h approaches zero. This equation is driven by the horizontal component of gravity (fourth term in Eq. (1.1)), the vertical component of gravity (third term in Eq. (1.1); represents second order diffusion) and surface tension (second term in Eq. (1.1), represents fourth-order diffusion). Here, the dimensionless parameters $D(\theta) = \epsilon \cot(\theta)$, is the ratio of the size of the vertical and the horizontal components of gravity, $Ca = (\epsilon^3\gamma)/(\mu U)$, is an inverse capillary number (compares surface tension to viscous forces) and θ is the inclination angle. Here, $U = (\rho g \sin(\theta) H^2)/\mu$ is a characteristic velocity based on balancing viscous forces with the horizontal component of gravity. These parameters depend on viscosity μ, the surface tension at the air liquid interface γ, acceleration due to gravity g and the fluid density ρ, all taken to be constant.

We first seek y-independent numerical solution of Eq. (1.1) for constant flux and constant volume boundary conditions (described later on) satisfying

$$h_t + Q_x = 0,\ Q = \frac{1}{3} Ca h^3 h_{xxx} - \frac{1}{3} D(\theta) h^3 h_x + \frac{1}{3} h^3, \tag{1.2}$$

where Q is the liquid flux. Analytical solutions to Eq. (1.2) are difficult to obtain for general boundary and initial conditions. One then needs to resort to numerical solution.

Numerical solutions of Eq. (1.2) have been done using finite difference schemes on a fixed uniform mesh (see Kondic [57], for example). The solution successfully capture the main characteristics of the evolution of $h(x,t)$, namely, the capillary ridge near the leading edge of the drop and capillary waves at both the leading and trailing edges of the drop (see illustration in figure 1.2). However, for Capillary number $Ca \ll 1$ (which corresponds to large surface tension) the width of the capillary ridge is much smaller (hence, the slope and curvature there becomes much larger). Hence, a uniform mesh finite difference scheme would require a large number of mesh points to accurately resolve the capillary ridge region resulting in a large overall number of mesh points and increased computational effort. We show in chapter 3 how this can be resolved using an adaptive moving mesh scheme.

For the two-dimensional problem in Eq. (1.1), we are interested in numerically simulating the fingering instability observed in figure 1.1. Linear stability analysis of this problem using the y-independent solution obtained from Eq. (1.2) as the base state have shown this solution to be linearly unstable to small-amplitude sinusoidal perturbations in the transverse y-direction above a critical angle θ from the horizontal (see Bertozzi & Brenner [7] and Kondic & Diez [58]). This critical angle has an implicit dependence on the parameters Ca and $D(\theta)$ but in generally around $3-5^o$ for physically realistic values of Ca and $D(\theta)$. Numerical solutions for the nonlinear fingering behaviour have been obtained using finite differences both on a uniform mesh and non-uniform (but fixed) mesh (Kondic & Diez [58, 57, 59]), using an alternate

-direction-implicit (ADI) time-stepping technique [40] and using both an adaptive finite element [77] and finite difference method [62] based on a multigrid method. One would require a large number of uniform mesh points in the y-direction to accurately resolve a finger. We show in chapter 5 how this can be effectively resolved using a two-dimensional adaptive moving mesh scheme.

We now consider the above problem with the inclusion of surfactant and surfactant-related effects. The bulk flow in the drop or sheet is governed by the Navier-Stokes equations. The surfactant is considered insoluble (i.e., it is present only on the free surface) and its transport on the free surface is modelled using an advection-diffusion equation. Lubrication theory can be applied to reduce the governing equations to give the evolution equations for the film thickness $h(x, y, t)$ and the surfactant concentration $\Gamma(x, y, t)$. The interested reader can refer to Edmonstone et al. [32, 36, 34, 38] for their derivation. These can be written as

$$h_t + \boldsymbol{\nabla} \cdot \left[\frac{1}{3} Ca h^3 \boldsymbol{\nabla} \nabla^2 h - \frac{1}{3} G \cos\theta h^3 \boldsymbol{\nabla} h - \frac{1}{2} h^2 \boldsymbol{\nabla} \Gamma\right] + \left[G \sin\theta \frac{1}{3} h^3\right]_x = 0, \quad (1.3)$$

$$\Gamma_t + \boldsymbol{\nabla} \cdot \left[\frac{1}{2} Ca h^2 \Gamma \boldsymbol{\nabla} \nabla^2 h - \frac{1}{2} G \cos\theta h^2 \Gamma \boldsymbol{\nabla} h - h\Gamma \boldsymbol{\nabla} \Gamma\right] + \left[G \sin\theta \frac{1}{2} h^2 \Gamma\right]_x = \frac{1}{Pe} \nabla^2 \Gamma. \quad (1.4)$$

The scalings used to derive Eq. (1.3) are different those used to derive Eq. (1.1). Here, the balance is between viscous forces and surface tension gradients (due to surfactant concentration gradients). The spreading of the drop and transport of surfactant are

driven by the horizontal component of gravity (fifth terms on left-hand side of Eqs. (1.3, 1.4)), the vertical component of gravity (third terms on left-hand side of Eqs. (1.3, 1.4)), surface tension (second terms on left-hand side of Eqs. (1.3, 1.4) and surface tension gradients related to surfactant concentration gradients (fourth terms on left-hand side of Eqs. (1.3, 1.4). The surface diffusion of surfactant also contributes to its transport (the right-hand side of Eq. (1.4)). Here, the dimensionless parameters $Ca = \epsilon^2\sigma_m/\Pi$, is an inverse capillary number (compares surface tension to viscous forces), $G = (\rho g H L)/\Pi$ is the Bond number (relating gravity and viscous forces), $Pe = (\Pi H)/(\mu D_s)$ is a Péclet number (compares the magnitude of changes to surfactant concentration due to diffusion and advection by surface flow) and θ is the inclination angle. $U = (\Pi H)/(\mu L)$ is the characteristic Marangoni velocity in this problem. These parameters depend on surface diffusivity D_s of the surfactant , viscosity μ, the spreading pressure Π (is used to scale the surface tension), acceleration due to gravity g the fluid density ρ and the maximal value of the surface tension σ_m, all taken to be constant.

We seek one-dimensional (y-independent) solutions of Eqs. (1.3, 1.4) for constant flux and constant volume boundary conditions (described later on) satisfying

$$h_t + Q_f = 0, \;\; Q_f = \boldsymbol{\nabla}\cdot\left[\frac{1}{3}Cah^3\boldsymbol{\nabla}\nabla^2 h - \frac{1}{3}G\cos\theta h^3\boldsymbol{\nabla}h - \frac{1}{2}h^2\boldsymbol{\nabla}\Gamma\right] \qquad (1.5)$$

$$+\left[G\sin\theta\frac{1}{3}h^3\right]_x,$$

$$\Gamma_t + Q_\Gamma = 0, \quad Q_\Gamma = \boldsymbol{\nabla} \cdot \left[\frac{1}{2} Ca h^2 \Gamma \boldsymbol{\nabla} \nabla^2 h - \frac{1}{2} G \cos\theta h^2 \Gamma \boldsymbol{\nabla} h - h \Gamma \boldsymbol{\nabla} \Gamma \right] \tag{1.6}$$
$$+ \left[G \sin\theta \frac{1}{2} h^2 \Gamma \right]_x - \frac{1}{Pe} \nabla^2 \Gamma,$$

where Q_f and Q_Γ are fluid and surfactant fluxes, respectively. Numerical solution of Eqs. (1.5,1.6) have been done using finite difference schemes on a fixed uniform mesh, collocation methods (using PDECOL and EPDCOL [56, 64]) and finite element methods [32, 36, 34, 38, 60]. The solution behaviour shows multiple wave-like structures in addition to the capillary ridge and capillary waves mentioned in the previous problem. These include a step-like structure ahead of the leading edge of the drop and a fluid front upstream of its trailing edge. The slope and curvature of these structures become much steeper for $Ca \ll 1$ and hence a scheme incorporating a uniform mesh would require a large number of mesh points to accurately resolve all these structures. We show in chapter 4 how by carefully selecting a monitor function representing region where the solution changes rapidly and incorporating it into an adaptive moving mesh scheme can accurately resolve these structures. In this thesis we only consider the 1D problem, Eqs. (1.5,1.6). Edmonstone *et al.* [32, 36, 34, 38] have also considered fully nonlinear simulations of Eqs. (1.3, 1.4)) to explore the stability of their spatially one-dimensional solutions of Eqs. (1.5,1.6) to transverse perturbations in the flow and surfactant concentrations. Their numerical simulations using a finite difference scheme on a uniform mesh and an ADI time-stepping scheme captures the fingering instability similar to the gravity-driven spreading problem.

Boundary conditions

We now prescribe boundary conditions for the gravity-driven spreading problem Eq. (1.1) and the gravity and surfactant-driven spreading problem Eqs. (1.3,1.4). Two sets of boundary conditions (BCs) are considered in this thesis, namely constant flux and constant volume. Constant flux BCs correspond to constant supply of fluid and surfactant concentration at one end of the domain while constant volume prescribes no flux BCs at both ends of the domain. We also assume that the plane is pre-wetted with a precursor film of thickness $b = b^*/H^* \ll 1$, which relieves the contact line singularity, and it is surfactant free. Here, b^* is the dimensional precursor film thickness and H^* is a characteristic film thickness. These boundary conditions are prescribed as follows:

The constant flux boundary conditions are given by

$$h_{xxx} = 0,\ h = 1,\ \text{at } x = 0, \quad h_{xxx} = 0,\ h = b,\ \text{at } x = L_x, \tag{1.7}$$

$$\Gamma = 1,\ \text{at } x = 0, \quad \Gamma = 0,\ \text{at } x = L_x, \tag{1.8}$$

where the computational domain in this case is given by $x \in [0, L_x]$. These correspond to constant supply of fluid (or constant flux of fluid) and surfactant concentration the end at $x = 0$. A schematic of this BC is shown in figure 1.5(a).

The constant volume boundary conditions are given by

$$h_{xxx} = 0,\ h = b,\ \text{at } x = \pm L_x, \quad \Gamma = 0,\ \text{at } x = \pm L_x. \tag{1.9}$$

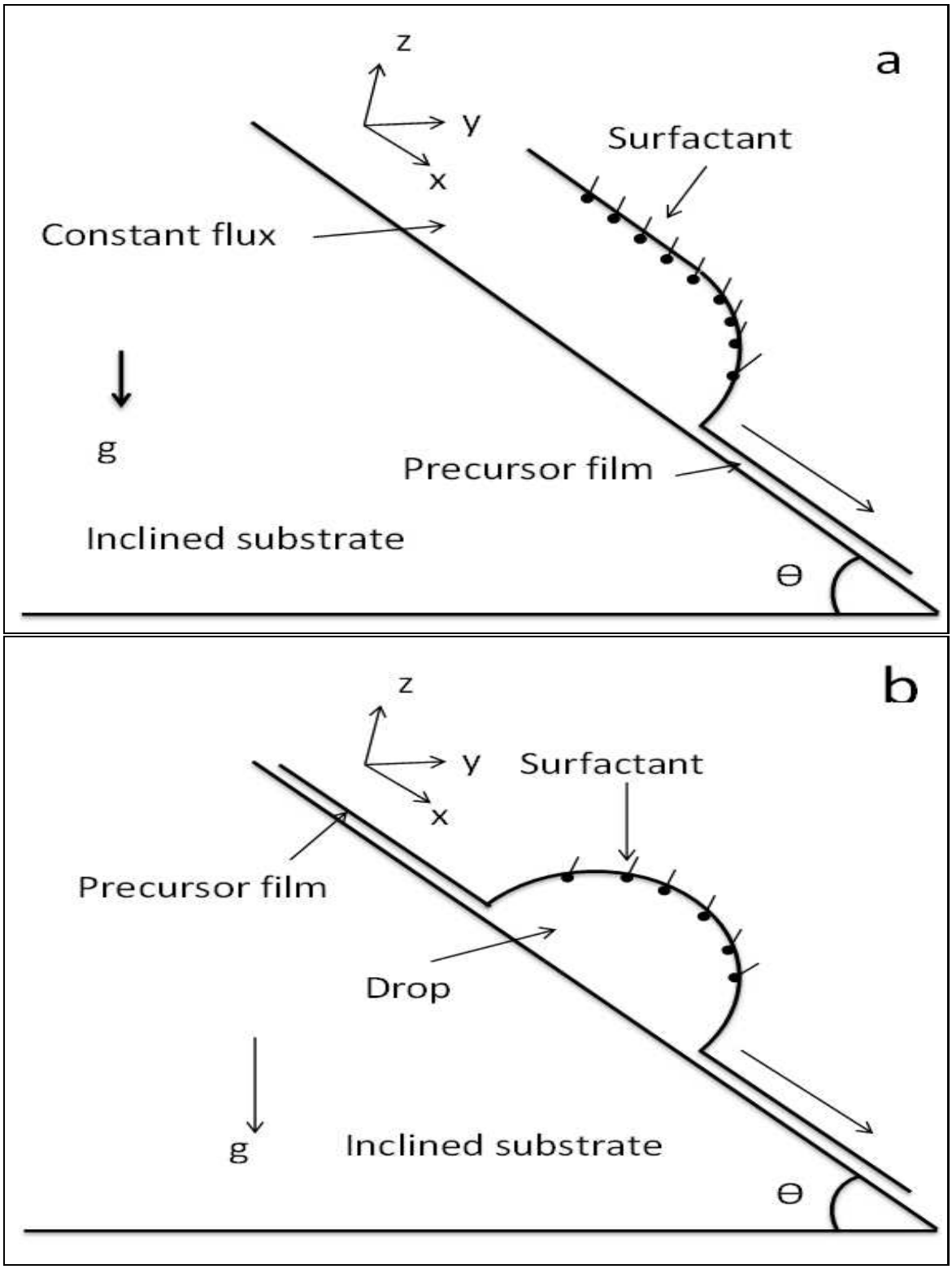

Figure 1.5: These figures illustrate a schematic of a surfactant-laden drop or sheet spreading on an inclined plane with (a) constant flux boundary conditions and (b) constant volume boundary conditions.

A schematic of this BC is shown in figure 1.5(b). The computational domain in this case is given by $x \in [-L_x, L_x]$.

For the two-dimensional gravity-driven spreading problem in Eq. (1.1) we apply symmetry boundary conditions in the y direction which are given by $h_y = h_{yyy} = 0$.

1.3 Adaptive mesh techniques

Numerical solution of partial differential equations are obtained by approximating the PDEs on a mesh by basically discretising the physical domain into a finite set of points. A uniform mesh is commonly used to discretise the physical domain (see figure 1.6(a) for an illustration of a solution on a uniform and fixed mesh). More mesh points are required to resolve the steep front, resulting in solving a large system of equations. This is computationally intensive and expensive. So, an alternative approach is to have an adaptive mesh which redistributes or puts more points in regions where the solution changes rapidly and less points elsewhere (see figure 1.6(b) for an illustration of this). Although discretisation schemes on a uniform mesh are routinely used for the numerical solution of thin film problems, over the past two decades or so adaptive moving mesh techniques have also been utilised in problems with rapid spatial variation such as, steep fronts, shock-like structures and moving contact lines [47]. The aim of an adaptive mesh is to obtain an acceptable degree of accuracy and efficiency without having an excessive number of mesh points, as compared to a uniform mesh. As pointed out by Brackbill [10],

"*the marginal utility of using an adaptive grid over using a uniform grid is much greater than the marginal utility of using one method for adaptive gridding over another. It is almost always better to use an adaptive grid of any kind than to use none.*"

Moreover, as illustrated in figures 1.2,1.4, the front is moving in time. So, not only is an adaptive mesh needed to accurately resolve the front, but it is also necessary to

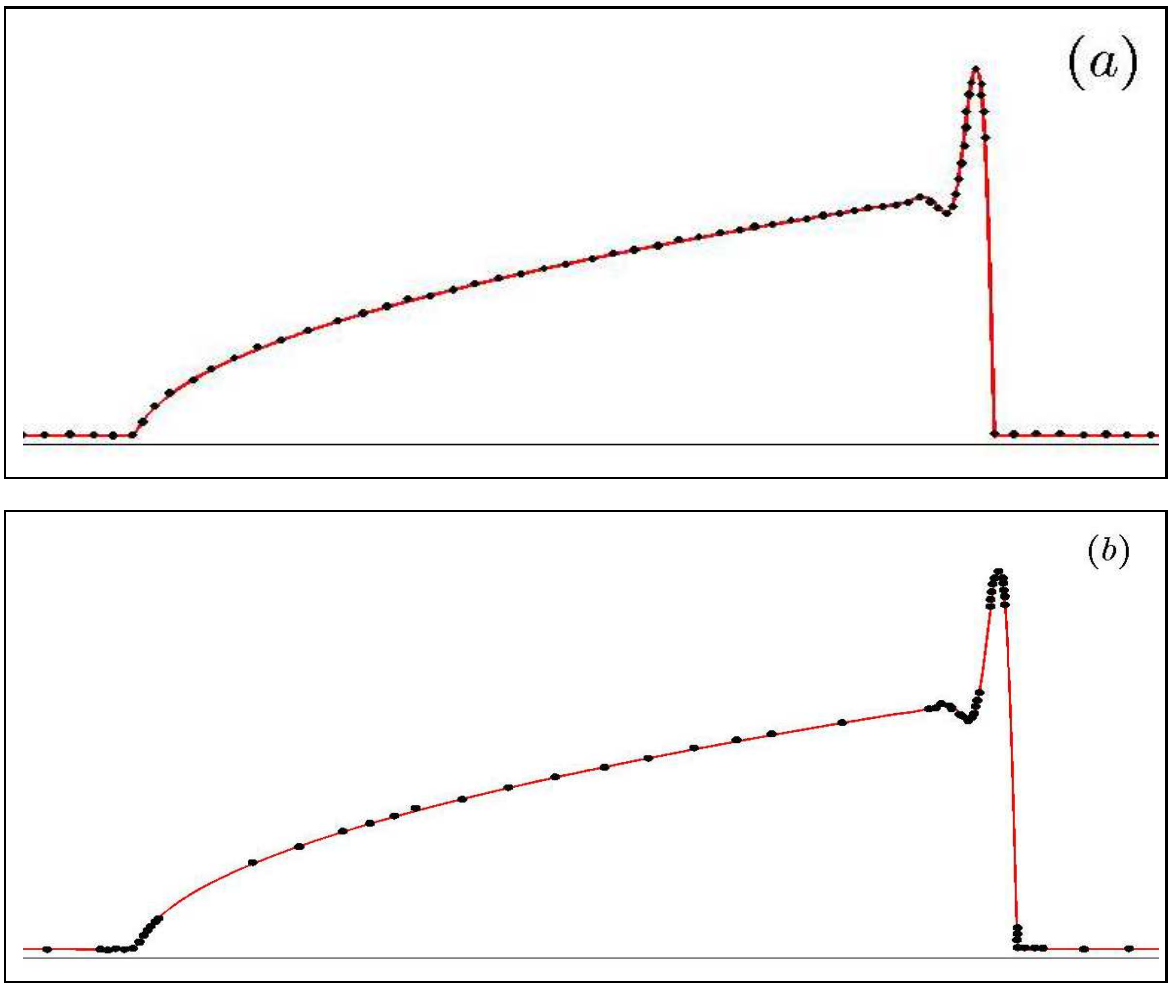

Figure 1.6: These figures illustrate the differences between (a) a uniform mesh and (b) an adaptive mesh. A large number of mesh points are needed to resolve the steep front using a uniform mesh. The adaptive mesh redistributes the mesh points so that more points are in regions where the solution has rapid variation and fewer points in regions where the solution varies slowly.

move the mesh points in time in order to conform with the moving front.

Many adaptive mesh strategies have been utilized for investigating the numerical solution of partial differential equations. We categorise these methods into three groups [78]:

h-refinement methods (or h-adaptive mesh methods): h-refinement methods are the most widely applied and developed of these methods and generally use the finite element method to discretise the underlying PDE. They form the basis of several commercial codes which usually start with a uniform mesh and then automatically coarsen or refine the mesh by the inclusion or deletion of mesh points. The strategy for applying this is usually based on some *a posteriori* indicator of the solution error. Barrett *et al* [4], Sun *et al* [77] and Li *et al* [62] have applied h-refinement techniques using the finite element method for fourth order degenerate parabolic PDEs in one and two dimensions. These h-refinement techniques form the basis of general purpose software packages for one-dimensional parabolic PDE systems, for example, BACOL (B-spline Adaptive COLlocation) [82] and D03PPF [70].

p-refinement (or p-adaptive mesh methods): In p-refinement methods a finite element discretisation of the PDEs is utilized with basis functions involving polynomials of a particular order. Then, based on some error estimate, this order is increased or decreased. p-refinement is usually never applied alone and is combined with h-refinement to generate more sophisticated techniques such as, hp-refinement. HPNEW [66] and HPDASSL [65] are general purpose software packages which use hp-refinement

for the solution of one-dimensional parabolic PDE systems.

r-refinement (or r-adaptive methods): These are also referred to as moving mesh methods. A moving mesh method has a fixed number of mesh points that are re-distributed based on a monitor function (could be based on variation in the solution gradient or curvature) and then the mesh is moved by solving the so-called moving mesh PDEs (MMPDEs) [41]. The MMPDEs are derived based on, for example, the equidistribution principle [18, 73] and variational methods [53, 13]. Both finite element and finite difference methods have been used to discretise the underlying PDE and MMPDEs. Although r-adaptive methods are a recent development and have not been used as frequently as h- or p-refinements, they have been successfully applied in various applications such as, computational fluid mechanics [78], convective heat transfer [24] and meterological [17, 81] problems. The r-refinement techniques also form the basis of general purpose software packages for one-dimensional parabolic PDE systems, for example, MOVCOL [50] and TOMS731 [9]. The interested reader can refer to the book, *Adaptive Moving Mesh Methods*, by Huang & Russell [54] and the review paper by Budd *et al.* [13], who have made substantial contributions to r-adaptive methods over the last twenty years.

The main aim of the thesis is to investigate existing r-adaptive or moving mesh strategies for the numerical solution of fourth order nonlinear degenerate parabolic PDEs in one and two dimensions. The goal is to determine the efficiency and accuracy of these techniques in comparison to uniform mesh schemes. We will use as test problems: a

single one-dimensional fourth order nonlinear parabolic PDE given by Eq. (1.2), a system of two coupled fourth order and second order parabolic one-dimensional PDEs given by Eqs. (1.5,1.6) and a single fourth order nonlinear parabolic PDE in two dimensions given by Eq. (1.1). While r-adaptive methods have been applied with success to second order nonlinear parabolic PDEs, to the best of our knowledge, they have never been investigated for fourth order parabolic PDEs. We emphasise that our goal is not to claim superiority of r-adaptive methods over hp-adaptive methods for this class of PDEs but rather an exploration of their potential usefulness.

1.4 Thesis outline

The outline of this thesis is organised as follows. In Chapter 2, we provide an overview of the main ingredients of r-adaptive or moving mesh methods based on the book, *Adaptive Moving Mesh Methods*, by Huang & Russell [54] and the review paper by Budd *et al.* [13]. We can ordinarily classify r-adaptive methods as either velocity-based methods [63, 22] or location-based methods [14, 48]. We focus on location-based methods. We define the 1D and 2D equidistribution principles, first used by de Boor [29] and Burchard [18]. The equidistribution principle is then used to derive one-dimensional MMPDEs. The variational method is also used to derive MMPDEs in both one and two dimensions. A good choice of the monitor function is an essential basis for the success of r-adaptive methods. We consider several monitor functions which can be obtained either by an error estimate or based on solution characteristics.

We also provide the general numerical framework followed in this thesis.

In Chapter 3, we investigate the numerical solution of a one-dimensional fourth order nonlinear parabolic PDE given by Eq. (1.2) using a uniform mesh and adaptive moving mesh scheme with a variety of MMPDEs and monitor functions. The numerical solutions are compared with a travelling wave solution that exists for constant flux BCs using which we determine an optimal MMPDE and monitor function for this problem. We also show for the constant volume case that the weights associated with the monitor function need to be adapted so as to redistribute the mesh points to several regions which require accurate resolution.

In Chapter 4, we investigate the numerical solution of a system of two one-dimensional fourth and second order parabolic PDEs using the uniform and the adaptive moving mesh scheme with a strategy informed by results from Chapter 3. We will see how to modify the monitor function so as to include solution characteristics of both components required for their overall accuracy.

In Chapter 5, we investigate the two-dimensional evolution of Eq. (1.1) using an adaptive moving mesh scheme with two-dimensional MMPDEs including the Parabolic-Mongé -Ampere (PMA) equation. We investigate the fingering instability that develops during the evolution and compare the numerical solutions to determine the efficiency and accuracy of the adaptive moving mesh schemes.

In Chapter 6 we discuss some future work that includes some thin film flow problems that pose challenges to the successful implementation of r-adaptive methods.

Chapter 2

Adaptive moving mesh methods: An Overview

In this chapter, we present an overview of the theoretical and practical aspects of adaptive mesh methods, with emphasis on its application for solving time-dependent PDEs discussed in Chapter 1. We focus on the theory of r-adaptive moving mesh methods which are a more recent development than hp-adaptive methods. The r-adaptive methods can be classified into two groups [23]. These are: location-based methods (control directly the redistribution of mesh point locations) [14, 48] and velocity-based methods [63, 22] (control the mesh velocity which on integration gives location of the mesh points). We only consider location-based methods in this thesis which are more widely used than velocity-based methods. In section §2.1 we consider the equidistribution principle, which is the underlying framework for the so-called moving mesh

Partial Differential Equation (MMPDE) methods considered in section §2.2. This section also considers MMPDEs based on variational methods. Section §2.3 considers the so-called optimal-transport-based methods, most notably the Monge-Ampère and the Parabolic Monge-Ampère (PMA) equation. We include discussion in both one and two dimensions of the above methods. In location-based methods, the local density of the mesh points is controlled by a so-called mesh-density function, often referred to as a monitor function. These are discussed in section §2.4. We then discuss in section §2.5 regularisation of MMPDEs and smoothing of monitor function required in their practical implementation. We emphasise here that the material presented in this chapter is adapted following the book, *Adaptive Moving Mesh Methods*, by Huang & Russell [54] and the review paper by Budd *et al.* [13], who have made substantial contributions to r-adaptive methods over the last twenty years.

2.1 The Equidistribution Principle

The concept of equidistribution plays an important role in formulating most of the adaptive moving mesh methods we discuss in this chapter. We first consider the idea behind equidistribution in one dimension.

Consider a given positive integer $N > 1$ and a positive continuous function $\rho = \rho(x)$ defined on an interval $[a, b]$. This continuous function $\rho(x)$ will subsequently be referred to as a monitor function. The equidistribution principle requires determining a mesh,

$\mathcal{J}_h$: $a = x_1 < x_2 < \ldots < x_{N+1} = b$, such that $\rho(x)$ is evenly distributed between the subintervals determined by these mesh points [3]. Hence,

$$\int_{x_1}^{x_2} \rho(x)dx = \int_{x_2}^{x_3} \rho(x)dx = \cdots = \int_{x_N}^{x_{N+1}} \rho(x)dx. \tag{2.1}$$

This means that the area under $\rho(x)$ is the same for each subinterval. Hence, the mesh points near the region(s) where $\rho(x)$ is large will be close together and will move further apart, where $\rho(x)$ is small. If $\mathcal{J}_h$ satisfies the relation Eq. (2.1) then it is called an

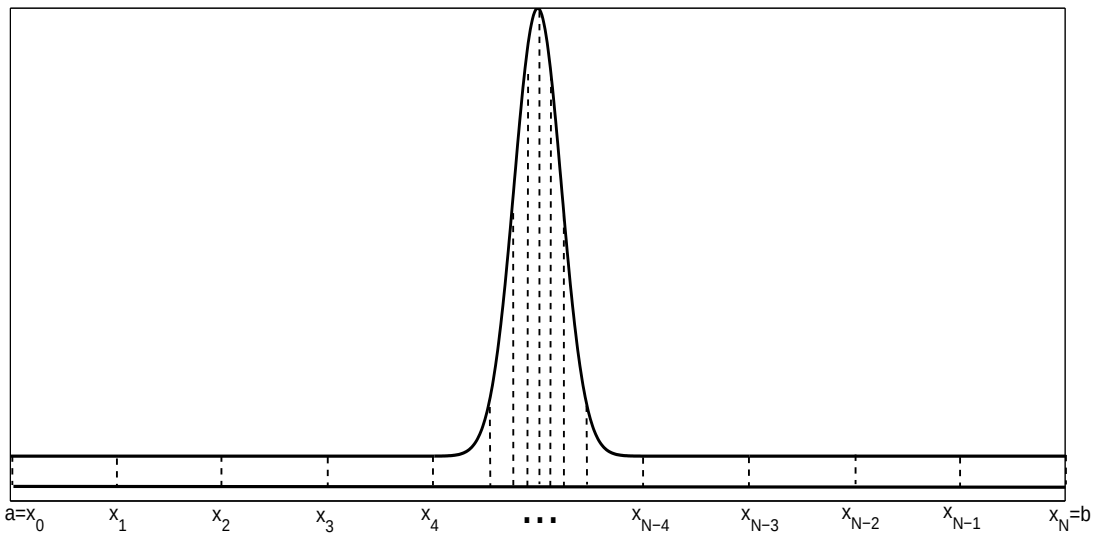

Figure 2.1: This figure shows an illustration of the equidistributing mesh that satisfies the relation Eq. (2.1).

equidistributing mesh for the monitor function $\rho(x)$ [53]. The objective of designing the monitor function is to control the concentration of the mesh points. Figure 2.1 shows an illustration of the equidistributing mesh that satisfies the relation Eq. (2.1). To practically implement Eq. (2.1) for the equidistributing mesh $\mathcal{J}_h$, it is more instructive

to consider this as a coordinate transformation:

$$x = x(\xi) \colon \Omega_c \equiv [0,1] \to \Omega_p \equiv [a,b],$$

where Ω_c and Ω_p are referred to as the computational and physical domains, respectively. Then,

$$x_j = x(\xi_j), \qquad j = 1, \ldots, N+1,$$

where

$$\xi_j = \frac{j-1}{N}, \qquad j = 1, \ldots, N+1,$$

is a uniform mesh on $[0,1]$.

The Eq. (2.1) can be written as

$$\int_a^{x_j} \rho(x)dx = \xi_j \theta, \qquad j = 1, \cdots, N+1, \tag{2.2}$$

$$\theta = \int_a^b \rho(x)dx. \tag{2.3}$$

In general, a continuous mapping $x = x(\xi)$ is called an equidistributing coordinate transformation for $\rho(x)$ if it satisfies:

$$\int_a^{x(\xi)} \rho(x)dx = \xi\ \theta, \qquad \forall \xi \in [0,1], \tag{2.4}$$

where θ is defined as above. Differentiating Eq. (2.4) once with respect to ξ, we obtain

$$\rho(x)\frac{dx}{d\xi} = \theta, \tag{2.5}$$

subject to the boundary conditions

$$x(0) = a, \qquad x(1) = b. \tag{2.6}$$

We see later that it is more useful in the practical sence to formulate the equidistributing coordinate transformation as an ODE in Eqs. (2.5,2.6) rather than solving the integral equation in Eq. (2.4). It can be shown that, for any strictly positive monitor function there exists a unique equidistributing coordinate transformation $x = x(\xi)$ for that $\rho(x)$. Although the existence and the uniqueness of the equidistributing mesh is guaranteed theoretically, it can rarely be determined analytically and in most cases needs to be approximated. A simple approximation for determining an equidistributing mesh is based on de Boor's algorithm [27, 28].

de Boor's algorithm Suppose that there is a known mesh density function on an arbitrary background mesh $\mathcal{J}_b$ such that

$$\mathcal{J}_b\colon a = z_0 < z_1 < z_2 < \ldots < z_M = b,$$

which can be thought of as a prescribed mesh in an iterative process. Here, M is used

to specify the number of points in the background mesh. Then, de Boor's algorithm approximates $p(x)$ on this background mesh by a piecewise constant function as follows

$$p(x) = \begin{cases} \frac{1}{2}(\rho(z_0) + \rho(z_1)) & \text{for } \; x \in [z_0, z_1] \\ \frac{1}{2}(\rho(z_1) + \rho(z_2)) & \text{for } \; x \in [z_1, z_2] \\ \cdots & \\ \frac{1}{2}(\rho(z_{M-1}) + \rho(z_M)) & \text{for } \; x \in [z_{M-1}, z_M] \end{cases} \tag{2.7}$$

and to then obtain the equidistributing mesh of N points (can be different from M) for this piecewise constant function. Denoting

$$P(x) = \int_a^x p(x)dx, \tag{2.8}$$

then, it can be obtained by

$$P(z_i) = \sum_{j=0}^{i} (z_{j+1} - z_j) \frac{\rho(z_{j+1}) + \rho(z_j)}{2}, \quad i = 0, \ldots, M. \tag{2.9}$$

Hence, the equidistribution relation Eq. (2.2) can be written as

$$P(x_i) = \xi_i \theta, \quad i = 0, \ldots, N \tag{2.10}$$

where θ is defined as $\theta = P(b)$. To obtain x_i, $i = 0, \ldots, N$, one first determines the

index k such that

$$P(z_{k-1}) \leq \xi_i \theta \leq P(z_k).$$

Since $\xi_i \leq 1$ and $P(x)$ should be piecewise linear, x_i is computed by

$$x_i = z_{k-1} + \frac{2(\xi_i P(z_k) - P(z_{k-1}))}{\rho(z_k) + \rho(z_{k-1})}.$$

Unfortunately, de Boor's algorithm requires to calculate Eq. (2.3) for every monitor function and does not extend easily to higher dimensions. A widely used method which can be generalised to higher dimensions is based on the ODE form given in Eqs. (2.5,2.6). Differentiating Eq. (2.5) again with respect to ξ, we have

$$\left(\rho(x)x_\xi\right)_\xi = 0, \qquad \xi \in [0,1], \tag{2.11}$$

subject to the boundary conditions

$$x(0) = a, \qquad x(1) = b. \tag{2.12}$$

We not that, the above equidistribution-related concepts are for time-independent problems. For the numerical solution of time-dependent problems, the monitor function depends on the solution and hence on time. So, the coordinate transformation will also depend on time, i.e., $x = x(\xi, t)$. One can still use the BVP in Eqs. (2.11,2.12) to determine $x = x(\xi, t)$ as follows:

$$(\rho(x,t)x_\xi)_\xi = 0, \qquad \xi \in [0,1], \tag{2.13}$$

subject to the boundary conditions

$$x(0) = a, \qquad x(1) = b. \tag{2.14}$$

It is considered advantageous to employ a PDE that involves a mesh speed rather than the ODE in Eq. (2.13). A mesh equation that involves a mesh speed is referred to as a moving mesh PDE (MMPDE) and discussed in section §2.2. Consider given positive integers $N_y > 1$ and $M_x > 1$ and a positive continuous function $\rho = \rho(\boldsymbol{x}, t)$ (where $\boldsymbol{x} = (x, y)$) on a rectangular domain $[a, b] \times [c, d]$. The equidistribution principle requires determining a mesh, $\mathcal{J}_h$: $a = x_1 < x_2 < \cdots < x_{M_x+1} = b, \quad c = y_1 < y_2 < \cdots < y_{N_y+1} = d$, such that

$$\int_{A_{j+\frac{1}{2},k+\frac{1}{2}}} \rho(\boldsymbol{x},t)d\boldsymbol{x} = \int_{A_{j+\frac{3}{2},k+\frac{1}{2}}} \rho(\boldsymbol{x},t)d\boldsymbol{x}, \text{ for } j = 2,\ldots,M_x-1,\ k = 2,\ldots,N_y, \tag{2.15}$$

where $A_{j+\frac{1}{2},k+\frac{1}{2}}$ denotes the mesh cell $[x_j, x_{j+1}] \times [y_k, y_{k+1}]$.

Consider this as a coordinate transformation:

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t) : \Omega_c \equiv [0,1] \times [0,1] \to \Omega_p \equiv [a,b] \times [c,d],$$

where $\boldsymbol{x} = (x, y)^T$ and $\boldsymbol{\xi} = (\xi, \eta)^T$. Then,

$$(\xi_j, \eta_k) = \left(\frac{j-1}{M_x}, \frac{k-1}{N_y} \right), \text{ for } j = 1, \ldots, M_x + 1, \ k = 1, \ldots, N_y + 1, \tag{2.16}$$

is a uniform mesh in $[0, 1] \times [0, 1]$. The relation Eq. (2.15) can written as

$$\int_a^{x_j} \int_c^{y_k} \rho(\boldsymbol{x}, t) d\boldsymbol{x} = \frac{j-1}{M_x} \frac{k-1}{N_y} \theta, \text{ for } j = 1, \ldots, M_x + 1, \ k = 1, \cdots, N_y + 1, \tag{2.17}$$

where

$$\theta = \int_a^b \int_c^d \rho(\boldsymbol{x}, t). \tag{2.18}$$

In general, a continuous mapping $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t)$ is called an equidistribution coordinate transformation for $\rho(\boldsymbol{x}, t)$ if it satisfies:

$$\int_a^{x(\boldsymbol{\xi})} \int_c^{y(\boldsymbol{\xi})} \rho(\boldsymbol{x}, t) d\boldsymbol{x} = \xi \eta \theta, \qquad \forall (\xi, \eta) \in [0, 1] \times [0, 1], \tag{2.19}$$

where θ is determined in Eq. (2.18). Since the above relationship is true for all (ξ, η), the coordinate transformation mesh must satisfy the identity

$$\rho(\boldsymbol{x}, t) \left| \frac{d\boldsymbol{x}}{d\boldsymbol{\xi}} \right| = \theta, \tag{2.20}$$

where $\dfrac{d\boldsymbol{x}}{d\boldsymbol{\xi}}$ is the Jacobian matrix of the transformation $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t)$, which is given by

$$\boldsymbol{J} = \begin{bmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{bmatrix}, \tag{2.21}$$

and $\left|\dfrac{d\boldsymbol{x}}{d\boldsymbol{\xi}}\right|$ is the determinant of the Jacobian, $J = |\boldsymbol{J}| = x_\xi\ y_\eta\ -\ x_\eta\ y_\xi$. Hence, Eq. (2.20), becomes

$$\rho(\boldsymbol{x}, t)J = \theta, \tag{2.22}$$

where θ is determined in Eq. (2.18). This problem does not have a unique solution and additional conditions are required to ensure existence and uniqueness of a mesh. This will be considered in §2.2.

2.2 Location-based adaptive moving mesh methods for time dependent problems

2.2.1 Moving mesh partial differential equations (MMPDEs)

In this section, we describe moving mesh methods based on MMPDEs, derived using the equidistribution principle. We first consider them in one dimension. We suppose the monitor function $\rho(x, t)$ is a given continuous function in x and t. Discretising the boundary value problem in Eq. (2.13) in the computational domain yields a system of

nonlinear equations which is difficult to solve directly. Alternatively, Huang, Ren and Russell [73] present a time evolution by taking the total time derivative of Eq. (2.13):

$$\frac{D\left(\rho x_\xi\right)_\xi}{Dt} = 0 \;\Rightarrow\; \left(\rho x_\xi\right)_{\xi t} + \frac{x_t}{x_\xi}\left(\rho x_\xi\right)_{\xi\xi} = 0. \tag{2.23}$$

where $(x_\xi, x_t) = \left(\frac{\partial x}{\partial \xi}, \frac{\partial x}{\partial t}\right)$ and $\frac{Df}{Dt} = \frac{\partial f}{\partial t} + x_t \frac{\partial f}{\partial x}$ is the total derivative of f with respect to t. Eq. (2.23) is referred to as MMPDE0. We obtain the equidistributing mesh by solving MMPDE0 starting with any initial mesh, e.g. uniform mesh. However, there is no guarantee that the initial mesh satisfies the equidistribution relation. This could lead to mesh tangling which can occur if $x_\xi < 0$, i.e., $x_j > x_{j+1}$. We can alleviate this problem by introducing a relaxation time into the solution of Eq. (2.23). One possible form is given by

$$\tau x_t = \left(\rho x_\xi\right)_\xi, \tag{2.24}$$

where $\tau > 0$ is a relaxation time. The equation Eq. (2.24) is referred to as MMPDE5 [73]. If τ is small, then the mesh will adapt quickly to changes in ρ, hence equidistributing quickly. For τ large then the mesh will adapt slowly to changes in ρ, hence equidistributing slowly. Once equidistributed, the mesh speed is zero. An alternative MMPDE equation can be determined from Eq. (2.5) by integrating both sides with respect to ξ and taking the residual:

$$R = \xi\theta(t) - \int_0^\xi \rho(x,t) x_\xi d\xi. \tag{2.25}$$

Setting $R = \tau_1 x_t$ and differentiating twice with respect to ξ on both sides, leads to

$$\tau_1 (x_t)_{\xi\xi} = -\left(\rho x_\xi\right)_\xi . \tag{2.26}$$

This equation is referred to as MMPDE6 [25, 51]. If we combine MMPDE5 in Eq. (2.24) and MMPDE6 in Eq. (2.26) with $\tau_1 = -\gamma_1 \tau$, we obtain

$$\tau \left(1 - \gamma_1 \partial_{\xi\xi}\right) x_t = \left(\rho x_\xi\right)_\xi , \tag{2.27}$$

where γ_1 is a smoothing parameter. It is chosen to control the smoothness of the mesh. We refer to Eq. (2.27) as regularised MMPDE5.

2.2.2 Variational methods

The variational method is an alternative approach for deriving MMPDEs. The advantage of this approach is that it can be extended relatively easily to higher dimensions. For time-dependent problems, the differential form of the one-dimensional equidistribution Eq. (2.13) can be written as

$$\rho(x)_x \left(x_\xi\right)^2 + \rho(x) x_{\xi\xi} = 0, \qquad x(0,t) = a,\ x(1,t) = b, \forall t > 0. \tag{2.28}$$

We define a functional corresponding to Eq.(2.28) as

$$I[x] = \frac{1}{2}\int_0^1 (\rho(x)x_\xi)^2\, d\xi. \tag{2.29}$$

Using calculus of variations, a minimiser $x = x(\xi)$ of this functional will satisfy the Euler-Lagrange equation given by

$$F_x - \left(F_{x_\xi}\right)_\xi = 0, \quad \text{where} \quad F(\xi, x, x_\xi) = \frac{1}{2}(\rho x_\xi)^2. \tag{2.30}$$

Therefore, we can determine the equidistributing coordinate transformation $x = x(\xi)$ by minimising the functional $I[x]$ in Eq. (2.29). For the time-dependent transformation $x = x(\xi, t)$, we can consider t to be a parameter such that $x(\xi, t) \to x(\xi)$ as $t \to \infty$. Then a minimiser of $I[x]$ in Eq. (2.29) can be obtained by solving the following equation, referred to as the gradient flow equation,

$$x_t = -\frac{P}{\tau}\frac{\delta I}{\delta x}, \tag{2.31}$$

where $\dfrac{\delta I}{\delta x}$ is the functional derivative of $I[x]$. $\tau > 0$ is a parameter for adjusting the mesh response time to changes in the monitor function $\rho(x, t)$ and P could be chosen as a function of ρ or as a differential operator (as shown below), which can take various forms as we shall see later. It can be shown that

$$\frac{\delta I}{\delta x} = F_x - \left(F_{x_\xi}\right)_\xi, \quad \text{where} \quad F(\xi, x, x_\xi) = \frac{1}{2}(\rho(x, t)x_\xi)^2. \tag{2.32}$$

Hence, the variational method determines a coordinate transformation $x = x(\xi, t)$ which satisfies the equidistribution condition Eqs. (2.13, 2.14) by minimising the functional $I[x]$ in Eq. (2.29).

By using Eq. (2.30), with $F\left(\xi, x, x_\xi\right) = \frac{1}{2}\left(\rho x_\xi\right)^2$, we can prove that the Euler-Lagrange equation of Eq. (2.29) is given by Eq. (2.13) as follows

$$F_x - \left(F_{x_\xi}\right)_\xi = 0 \quad \Rightarrow \quad \rho x_\xi \rho_x x_\xi - ((\rho x_\xi)_\xi \rho + \rho x_\xi \rho_x x_\xi) = 0. \tag{2.33}$$

$$F_x - \left(F_{x_\xi}\right)_\xi = -\rho(\rho x_\xi)_\xi = 0. \tag{2.34}$$

Since, the monitor function $\rho > 0$, then

$$(\rho x_\xi)_\xi = 0.$$

Hence, we obtain the Euler-Lagrange equation Eq. (2.13) for $I[x]$. From Eq. (2.32) and Eq. (2.34), the gradient flow equation Eq. (2.31) (which is similar to the heat or differential equation) becomes

$$x_t = \frac{P}{\tau}\rho(\rho x_\xi)_\xi. \tag{2.35}$$

By choosing $P = \frac{1}{\rho}$ in Eq.(2.35), gives MMPDE5:

$$\tau x_t = \left(\rho x_\xi\right)_\xi. \tag{2.36}$$

Other choices are:

- Modified MMPDE5: $P = \frac{1}{\rho^2}$, leads to

$$x_t = \frac{1}{\tau\rho}\left(\rho x_\xi\right)_\xi . \tag{2.37}$$

- MMPDE4: $P = -\left(\frac{\partial}{\partial\xi}\rho\frac{\partial}{\partial\xi}\right)^{-1}\frac{1}{\rho}$, leads to

$$\left(\rho\ x_{t\xi}\right)_\xi = -\frac{1}{\tau}\left(\rho\ x_\xi\right)_\xi . \tag{2.38}$$

- MMPDE6: $P = -\left(\frac{\partial^2}{\partial\xi^2}\right)^{-1}\frac{1}{\rho}$, leads to

$$x_{t\xi\xi} = -\frac{1}{\tau}\left(\rho\ x_{t\xi}\right)_\xi . \tag{2.39}$$

Alternatively, it is convenient to consider $\xi = \xi(x)$ instead of $x = x(\xi, t)$ which satisfies

$$\left(\frac{1}{\rho}\xi_x\right)_x = 0. \tag{2.40}$$

In one dimension, equidistributing the monitor function is precisely equivalent to the Euler-Lagrange equation for the functional

$$I[\xi] = \frac{1}{2}\int_a^b \frac{1}{\rho}\left(\frac{\partial\xi}{\partial x}\right)^2 dx. \tag{2.41}$$

The gradient flow equation is given by

$$\xi_t = \frac{P}{\tau}\left(\frac{1}{\rho}\xi_x\right)_x . \tag{2.42}$$

Differentiating both sides of the relation $\xi = \xi(x(\xi,t),t)$ once with respect to t (keeping ξ fixed) and with respect ξ (keeping t fixed), we obtain

$$\xi_x = x_\xi^{-1}, \qquad \xi_t = -\xi_x x_t.$$

Substituting into Eq. (2.42), we obtain the mesh equation for $x = x(\xi,t)$ as

$$x_t = \frac{1}{\tau} x_\xi P \left(\rho x_\xi\right)^{-2} \left(x_\xi\right)^{-1} \left(\rho x_\xi\right)_\xi . \tag{2.43}$$

Choosing $P = \left(\rho x_\xi\right)^2$ in Eq.(2.43), gives MMPDE5:

$$\tau x_t = \left(\rho\ x_\xi\right)_\xi . \tag{2.44}$$

Other choices are:

- Modified MMPDE5: $P = \left(\rho x_\xi\right)^2 \rho^{-1}$, leads to

$$x_t = \frac{1}{\tau\rho}\left(\rho\ x_\xi\right)_\xi . \tag{2.45}$$

- MMPDE4: $P = -x_\xi^{-1}\left(\frac{\partial}{\partial\xi}\rho\frac{\partial}{\partial\xi}\right)^{-1}(\rho x_\xi)^2 x_\xi$, leads to

$$(\rho\ x_{t\xi})_\xi = -\frac{1}{\tau}(\rho\ x_\xi)_\xi . \tag{2.46}$$

- MMPDE6: $P = -x_\xi^{-1}\left(\frac{\partial^2}{\partial\xi^2}\right)^{-1}(\rho x_\xi)^2 x_\xi$, leads to

$$x_{t\xi\xi} = -\frac{1}{\tau}(\rho\ x_\xi)_\xi . \tag{2.47}$$

MMPDE5 and modified MMPDE5 given in Eqs. (2.44,2.45), respectively, are generally quite stiff and in practice it is proposed to regularise this and we refer to it as Regularised MMPDE5:

$$\tau\left(1-\gamma_1\partial_{\xi\xi}\right)x_t = (\rho\ x_\xi)_\xi . \tag{2.48}$$

Regularised modified MMPDE5:

$$\tau\left(1-\gamma_1\partial_{\xi\xi}\right)x_t = \frac{1}{\rho}(\rho\ x_\xi)_\xi . \tag{2.49}$$

A smoothing parameter γ_1 is related to the monitor function ρ (see Budd et al. [13] and references therein). An important advantage over Regularised MMPDE5 Eq. (2.48) is that Eq. (2.49) is more robust usually leading to stable meshes which extends easily to higher dimensions. It is essentially related to the PMA moving mesh method based on optimal transport which we discuss later on in this chapter.

The one dimensional functional Eq. (2.41) can be generalised to two dimensions as

$$I[\boldsymbol{\xi}] = \frac{1}{2}\int_{\Omega_p} \left(\nabla\xi^T\rho^{-1}\nabla\xi + \nabla\eta^T\rho^{-1}\nabla\eta\right) d\boldsymbol{x}, \tag{2.50}$$

where $\nabla = \left(\dfrac{\partial}{\partial x}, \dfrac{\partial}{\partial y}\right)$, and the monitor function $\rho(\boldsymbol{x}, t)$ is a 2×2 symmetric positive definite matrix [52, 21, 20]. The gradient flow equations are given by

$$\xi_t = \frac{1}{\tau\sqrt{\det(\rho\left(\boldsymbol{x}, t\right))}}\left(\nabla.\rho^{-1}\nabla\xi\right), \qquad \eta_t = \frac{1}{\tau\sqrt{\det(\rho\left(\boldsymbol{x}, t\right))}}\left(\nabla.\rho^{-1}\nabla\eta\right), \tag{2.51}$$

Using some transformation relations in Eq. (2.51), the following equations for the mesh can be written

$$\begin{aligned} x_t =& \frac{x_\xi}{\tau J\sqrt{\det(\rho)}}\left[\left(\frac{1}{J\det(\rho)}\left(x_\xi^T\rho x_\eta\right)\right)_\eta - \left(\frac{1}{Jdet(\rho)}\left(x_\eta^T\rho x_\eta\right)\right)_\xi\right] \\ &+ \frac{x_\eta}{\tau J\sqrt{det(\rho)}}\left[\left(\frac{1}{J\det(\rho)}\left(x_\eta^T\rho x_\xi\right)\right)_\xi - \left(\frac{1}{J\det(\rho)}\left(x_\xi^T\rho x_\xi\right)\right)_\eta\right], \end{aligned} \tag{2.52}$$

$$\begin{aligned} y_t =& \frac{y_\xi}{\tau J\sqrt{\det(\rho)}}\left[\left(\frac{1}{J\det(\rho)}\left(y_\xi^T\rho y_\eta\right)\right)_\eta - \left(\frac{1}{Jdet(\rho)}\left(y_\eta^T\rho y_\eta\right)\right)_\xi\right] \\ &+ \frac{y_\eta}{\tau J\sqrt{det(\rho)}}\left[\left(\frac{1}{J\det(\rho)}\left(y_\eta^T\rho y_\xi\right)\right)_\xi - \left(\frac{1}{J\det(\rho)}\left(y_\xi^T\rho y_\xi\right)\right)_\eta\right]. \end{aligned} \tag{2.53}$$

Here, $\det(\rho)$ is the determinant of the monitor function. These mesh equations can be easily discretised to move the mesh. The interested reader can refer to [13, 48]. The main disadvantage of solving the discretisations of Eqs. (2.52, 2.53) are that the resulting system is a highly nonlinear which is difficult to solve. Another disadvantage is that the resulting system requires the solution of a lower dimensional mesh equation

on the boundaries to obtain Dirichlet conditions for solving Eqs. (2.52, 2.53). The main advantage is that, under certain conditions on the monitor function, uniqueness and existence of the solution are ensured in two dimensions [81]. Alternatively, there is a method which derives mesh equations by solving a gradient flow problem related to a variational principle. This method is so-called Winslow method [25, 85]. The mesh equations can be written in the computational domain as

$$\tau x_t = \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\rho(\boldsymbol{x}, t)\boldsymbol{\nabla}_{\boldsymbol{\xi}} x)\,, \qquad \tau y_t = \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\rho(\boldsymbol{x}, t)\boldsymbol{\nabla}_{\boldsymbol{\xi}} y)\,. \tag{2.54}$$

Here, $\boldsymbol{\nabla}_{\boldsymbol{\xi}} = (\partial\xi, \partial\eta)^T$ and $\rho(\boldsymbol{x}, t)$ is a scalar function. This is the two-dimensional version of MMPDE5. Eq. (2.54) is significantly easier to solve than Eqs. (2.52, 2.53) and can also be easily discretised. However, discretisation of Eqs. (2.54) results in a very stiff system. Ceniceros and Hou [25] apply a low-pass filter smoothing to $\rho(\boldsymbol{\xi}, t)$. The low-pass filter can be explicitly applied to the mesh equation which is given by

$$\tau(1 - \gamma_1 \nabla^2_{\boldsymbol{\xi}}) x_t = \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\rho(\boldsymbol{x}, t)\boldsymbol{\nabla}_{\boldsymbol{\xi}} x)\,, \qquad \tau(1 - \gamma_1 \nabla^2_{\boldsymbol{\xi}}) y_t = \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\rho(\boldsymbol{x}, t)\nabla_{\boldsymbol{\xi}} y)\,, \tag{2.55}$$

where τ is the relaxation time parameter and γ_1 is a positive real number related to $\rho(\boldsymbol{x}, t)$. It is usually given by $\gamma_1 = \sqrt{\Delta t \, \max(\rho)}$ if a time step Δt is used or $\gamma_1 = \sqrt{\max(\rho)}$ if not. Eq. (2.55) is the two-dimensional version of regularised MMPDE5.

We can extend the one-dimensional MMPDE4 and MMPDE6 described previously to also obtain their two-dimensional versions:

2D MMPDE4:

$$\tau \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\rho \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}_t) = -\boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\rho \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}). \tag{2.56}$$

2D MMPDE6:

$$\tau \nabla^2_{\boldsymbol{\xi}} \boldsymbol{x}_t = -\boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\rho \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}). \tag{2.57}$$

The disadvantages in using the above MMPDEs are that we need to solve two PDEs coupled to the underlying PDE(s) to obtain the adaptive moving mesh and, more importantly, these MMPDEs cannot ensure no mesh tangling and acceptable mesh regularity. These disadvantages can be overcome by the optimal transport equations discussed next.

2.3 Optimal transport equations

Consider the computation domain $\Omega_c \equiv [0,1] \times [0,1]$, with the computational coordinates $\boldsymbol{\xi} = (\xi, \eta)$ continuously mapped to the physical domain $\Omega_p \equiv [0, L_x] \times [0, L_y]$, with the physical coordinates $\boldsymbol{x} = (x, y)$ such that

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t), \quad \boldsymbol{x} \in \Omega_p, \quad \text{and} \quad \boldsymbol{\xi} \in \Omega_c, \tag{2.58}$$

where L_x and L_y are the length of the domain in x and y directions, respectively. In two dimensions, this mapping does not ensure existence and uniqueness of the solution only based on the equidistribution principle. This results in mesh tangling and loss of

mesh regularity as was discussed in §2.2. Alternatively, we consider a mesh generation method based on optimal transportation. Optimal transport methods are a generalisation of one-dimensional mesh equations discussed in §2.2. To avoid the possibility of mesh tangling, Budd and Williams [15] propose that a good equidistributed mesh is near a uniform mesh. Recall from §2.2 the two-dimensional equidistribution principle Eq. (2.22) given by

$$\rho(\boldsymbol{x},t)|\boldsymbol{J}| = \theta(t), \quad \text{where} \quad \theta = \int_{\Omega_p} \rho(\boldsymbol{x},t)d\boldsymbol{x}, \tag{2.59}$$

and $\boldsymbol{J}$ is the Jacobian matrix is defined in Eq. (2.21). Brenier [11], Caffarelli [19] and Finn [42] prove that there is a unique optimal mapping $\boldsymbol{x}$ satisfying Eq. (2.59). This map $\boldsymbol{x}$ is determined by the gradient of a convex mesh potential $Q(\boldsymbol{\xi},t)$, such that

$$\boldsymbol{x}(\boldsymbol{\xi},t) = \nabla_{\boldsymbol{\xi}} Q(\boldsymbol{\xi},t), \tag{2.60}$$

where $\nabla_{\boldsymbol{\xi}} = (\dfrac{\partial}{\partial\xi}, \dfrac{\partial}{\partial\eta})^T$.

Substituting into the definition of the Jacobian matrix Eq. (2.21), leads to

$$\boldsymbol{J} = \begin{bmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{bmatrix} = \begin{bmatrix} Q_{\xi\xi} & Q_{\eta\xi} \\ Q_{\xi\eta} & Q_{\eta\eta} \end{bmatrix} = \boldsymbol{H}, \quad H = |\boldsymbol{H}| = Q_{\xi\xi}\, Q_{\eta\eta} \; - \; Q_{\xi\eta}^2, \tag{2.61}$$

where $\boldsymbol{H}$ is referred to as the Hessian matrix of $Q(\boldsymbol{\xi}, t)$ and H is its determinant. Hence, the two dimensional equidistribution principle Eq. (2.59) can be generalised to

$$\rho(\boldsymbol{x}, t)|\boldsymbol{H}| = \theta(t), \quad \text{where,} \quad \theta = \int_{\Omega_p} \rho(\nabla_{\boldsymbol{\xi}} Q, t) d\boldsymbol{x}, \tag{2.62}$$

Eq. (2.62) is referred to as the Monge-Ampère equation. The interested reader can refer to Budd *et al.* [16, 17].

A temporal relaxation method of Eq. (2.62) is proposed by Budd and Williams [16] to solve Eq. (2.62), which is given in the computational domain by

$$\tau \left(1 - \gamma_1 \nabla^2_{\boldsymbol{\xi}}\right) Q = \left(\rho(\boldsymbol{x}, t)|\boldsymbol{H}|\right)^{1/2}, \tag{2.63}$$

This equation is referred to as the Parabolic Monge-Ampère (PMA) equation. The smoothing operator $(1 - \gamma_1 \nabla^2_{\boldsymbol{\xi}})$ is similar to that used in §2.2, which is used to reduce the stiffness of the PDEs. The power 1/2 in the above equation is called the scaling power which is important for global existence of the solution and guarantees that the right hand side of eq. (2.63) scales linearly with $Q(\boldsymbol{\xi}, t)$ [17]. Again, τ is the relaxation time parameter and γ_1 is a positive real number related to $\rho(\boldsymbol{x}, t)$. It is usually given by $\gamma_1 = \sqrt{\Delta t \max(\rho)}$ if a time step Δt is used or $\gamma_1 = \sqrt{\max(\rho)}$ if not. The PMA equation Eq. (2.63) is subject to the Neumann boundary conditions:

$$Q_\xi = 0, L_x, \ \text{ at } \ \xi = 0, 1, \qquad Q_\eta = 0, L_y, \ \text{ at } \ \eta = 0, 1. \tag{2.64}$$

The initial condition for Eq. (2.63) is generally based on a uniform mesh chosen as

$$Q(\boldsymbol{\xi}, t = 0) = \frac{1}{2}\left(L_x\xi^2 + L_y\eta^2\right), \qquad \boldsymbol{\xi} \in \Omega_c. \tag{2.65}$$

The advantage of the PMA equation over the MMPDEs is that there is one equation less to solve in two dimensions to obtain the mesh and the meshes are generally regular and there is no mesh tangling.

2.4 Monitor function

The choice of the monitor function ρ is essential for the success of adaptive moving mesh methods. The monitor function can be chosen based on error estimates (e.g., based on polynomial interpolation or truncation error of a finite difference discretisation of a differential equation) or based on the solution characteristics of the underlying PDE. In the latter case, the monitor function can be defined by the solution u of the underlying PDE and possibly its derivatives so that

$$\rho(\boldsymbol{x}, t) = \rho(\boldsymbol{x}, u(\boldsymbol{x}, t), \boldsymbol{\nabla} u(\boldsymbol{x}, t), \nabla^2 u(\boldsymbol{x}, t), \cdots, t). \tag{2.66}$$

We just mention here several choices of the monitor function ρ that are commonly used, (see Appendix A for a selection of the monitor functions and Huang & Russell [54] for details of their derivation based on interpolation error estimates).

- In one-dimension, the arc-length monitor function is given by

$$\rho(x,t) = \sqrt{1 + |u_x|^2}. \tag{2.67}$$

In two dimensions, a scalar version of the one-dimensional arc-length monitor function generalised to two dimensions is given by

$$\rho(x,t) = \sqrt{1 + |\boldsymbol{\nabla} u(\boldsymbol{x},t)|^2}. \tag{2.68}$$

This monitor function is used by Ceniceros and Hou [24], and Tang *et al.* [86] for several applications in fluid mechanics.

- In one-dimension, the curvature-based monitor function is given by

$$\rho(x,t) = \sqrt{1 + \alpha\,|u_{xx}|^2}. \tag{2.69}$$

In two dimensions, a scalar version of the one-dimensional curvature monitor function generalised to two dimensions is given by

$$\rho(x,t) = \sqrt{1 + \alpha|\nabla^2 u(\boldsymbol{x},t)|^2}. \tag{2.70}$$

Here, α is a non-negative adaptivity parameter (or weight parameter) of the monitor function [5, 51] which controls the distribution of mesh points between regions where curvature changes rapidly and where curvature is almost negligible.

If $\alpha = 0$, then the monitor function $\rho = 1$ which represents a uniform mesh. For $\alpha > 0$, the monitor function ρ responds to changes in curvature and adapts the mesh points accordingly. Generally α is chosen to be a constant (equal to one, say). However, if there are regions of varying degrees of curvature, one would need to vary α with respect to x for the monitor function to capture these accurately.

- In one-dimension, the curvature-based monitor function is also given by

$$\rho(x,t) = \left(1 + \alpha \left|u_{xx}\right|^2\right)^{1/4}, \tag{2.71}$$

And its generalisation to two-dimensions is given by

$$\rho(x,t) = \left(1 + \alpha |\nabla^2 u(\boldsymbol{x},t)|^2\right)^{1/4}. \tag{2.72}$$

This monitor function is provided by Huang & Russell [54].

- An optimal monitor function [54] is also used in one-dimensional problems and is given by

$$\rho(x,t) = \left(1 + \frac{1}{\alpha}\left|u_{xx}\right|^2\right)^{1/3}, \quad \alpha = \left[\frac{1}{a-b}\int_a^b |u_{xx}|^2 dx\right]. \tag{2.73}$$

2.5 Monitor function smoothing and regularisation

If the solution of the underlying PDE is not smooth, the discretised monitor function can change abruptly and slow down the computation. To obtain a smoother mesh and also make the MMPDEs easier to integrate, it is common practice in the context of moving mesh methods to smooth the monitor function [31, 43]. Dorfi and Drury [31] propose a technique to smooth a one-dimensional mesh concentration. in this technique, if $n_i = (x_{i+1} - x_i)^{-1}$, then a smoother mesh is obtained by $\hat{n_i} = n_i - \gamma\,(n_{i+1} - 2n_i + n_{i-1})$. This technique is often called the Dorfi and Drury method. Another method to obtain a smoother mesh can be generated by averaging the monitor function ρ over all mesh points prior to the mesh adaptation [80]. In one-dimension, assume that the monitor function $\rho(x,t)$ is given, then the values of the smoothed ρ at the mesh points are given by

$$\hat{\rho}_i = \sqrt{\frac{\sum_{k=i-p}^{i+p} \rho_k^2 \left(\gamma/(1-\gamma)\right)^{|k-i|}}{\sum_{k=i-p}^{i+p} \left(\gamma/(1-\gamma)\right)^{|k-i|}}}, \tag{2.74}$$

where $\gamma \in (0,1)$ is called the smoothing parameter and $p \geq 0$ is referred to as the smoothing index. In two-dimensions, this can be written as

$$\hat{\rho}_{i,j} = \frac{\sum_{k=-1}^{+1} \sum_{l=-1}^{+1} \rho_{k+i,j+l} \gamma^{|k|+|i|}}{\sum_{k=-1}^{+1} \sum_{l=-1}^{+1} \gamma^{|k|+|i|}}. \tag{2.75}$$

For example, if $\gamma = \frac{1}{2}$ then Eq. (2.75) is given by

$$\begin{aligned}\hat{\rho}_{i,j} \leftarrow \quad & \tfrac{1}{4}\rho_{i,j} + \tfrac{1}{8}\left(\rho_{i+1,j} + \rho_{i-1,j} + \rho_{i,j+1} + \rho_{i,j-1}\right) \\ & + \tfrac{1}{16}\left(\rho_{i+1,j+1} + \rho_{i+1,j-1} + \rho_{i-1,j-1} + \rho_{i-1,j+1}\right).\end{aligned} \tag{2.76}$$

This is widely applied in the adaptive moving mesh literature [13].

2.6 Computation using adaptive moving mesh methods

Computing numerical solutions of the underlying PDE using an adaptive moving mesh method has three related problems:

1. The strategy used to adapt and move the mesh.

   As described in this chapter the mesh is moved by solving the moving mesh equations either MMPDEs or the PMA equation. We need to choose a suitable relaxation parameter τ and smoothing parameter γ_1 to maintain mesh regularity and avoid mesh tangling. The adaptivity of the mesh is controlled by a suitable choice of a monitor function. In this thesis we use the curvature monitor function Eqs. (2.69,2.70) and the arc length monitor function Eqs. (2.67,2.68). Smoothing of the monitor function may be required to obtain a smoother mesh and also make the MMPDEs easier to integrate

2. The method used to discretise the underlying PDE(s) and the mesh equation(s).

   The finite difference method is used to discretise the equations. We semi-discretise the spatial derivatives using second-order accurate finite differences and we keep the time derivative continuous. The resulting system of ODEs are then solved using the method of lines (see Morton & Mayers [67] or Smith [76] for finite difference techniques to solve PDEs).

3. The method used to solve the discretised system of equations.

   We solve the resulting ODEs for the semi-discretised PDE(s) and mesh equation(s) simultaneously using the stiff ODE solver DASSL [72] or DASPK [12]. These solvers use either a direct (DASSL) or iterative (DASPK - based on preconditioned Krylov subspace method) method to solve the linearised system of equations. They also allow approximating the Jacobian using an Incomplete LU factorisation which, if carefully done, can enhance the performance of the solver in comparison to the direct method.

Chapter 3

Numerical Solution of a 1D Fourth Order Parabolic PDE on an Adaptive Mesh

3.1 PDE with constant flux boundary conditions

Consider the one-dimensional fourth order degenerate nonlinear paraboic PDE described in Chapter 1,

$$h_t + \left[Ca\frac{h^3}{3}h_{xxx} - D(\theta)\frac{h^3}{3}h_x + \frac{h^3}{3} \right]_x = 0, \quad (x,t) \in [0,L] \times [0,T_f]. \tag{3.1}$$

Here, $h = h(x,t)$ is the film thickness, x is the spatial variable and t is time. The dimensionless parameters are $Ca \ll 1$, the Capillary number, $D(\theta)$ is related to the vertical component of gravity, θ is the inclination angle, L is a domain length and T_f is a specific time.

Equation (3.1) is supplemented by four boundary conditions (BCs) in x. These are:

$$h(0,t) = 1, \;\; h_{xxx}(0,t) = 0, \quad h(L,t) = b, \;\; \text{and} \;\; h_{xxx}(L,t) = 0. \tag{3.2}$$

Here, $b \ll 1$, is the precursor film thickness. This represents a constant flux of fluid introduced at the upstream end $x = 0$ and the substrate is pre-wetted with a thin precursor film of thickness b. The initial condition is chosen as

$$h(x,0) = (1 - x^2)\; H(1-x) + b\; H(x-1), \quad x \in [0, L], \tag{3.3}$$

where $H(x)$ is the Heaviside function. This represents a 1D drop connecting onto a precursor film of thickness b. Figure 3.1 shows this initial condition with $b = 10^{-2}$.

We seek the numerical solution of Eqs. (3.1-3.3) to determine the evolution of the film thickness $h(x,t)$ using the Finite Difference technique and the Method of Lines. A travelling wave solution exists for this problem [8, 57]. This is then used as an exact solution to compare the convergence and accuracy of the numerical solution on a uniform and adaptive moving mesh.

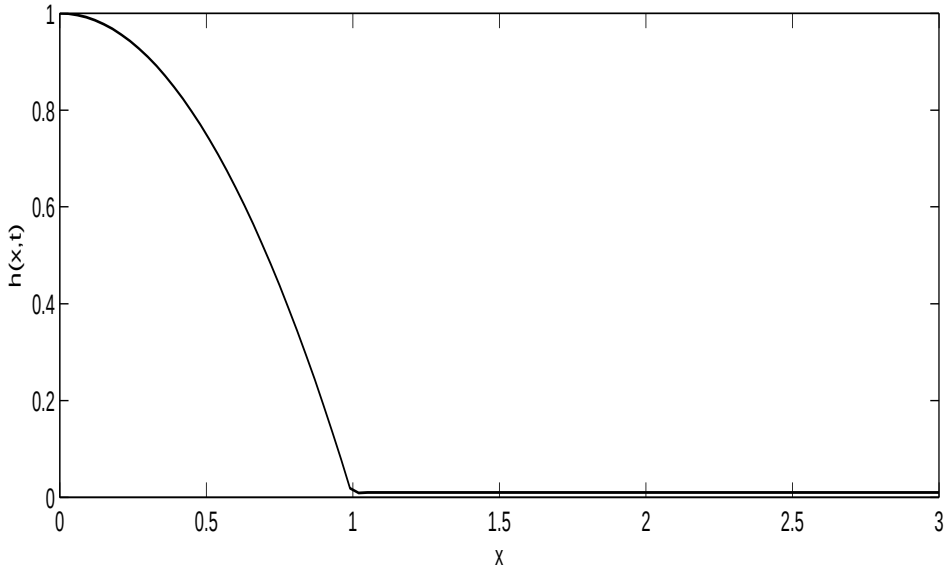

Figure 3.1: The initial condition for the film thickness h using Eq. (3.3). The precursor thickness $b = 10^{-2}$.

3.1.1 Travelling wave solution

We seek a travelling wave solution of Eqs. (3.1,3.2). We write,

$$\xi = \frac{(x - ct)}{Ca^{\frac{1}{3}}}, \quad h(x,t) = \hat{h}(\xi), \tag{3.4}$$

where c is the wave speed. Substituting these scalings into Eq. (3.1), leads to

$$-c\ \hat{h}_\xi + \left[\frac{\hat{h}^3}{3}\ \hat{h}_{\xi\xi\xi} - \hat{D}(\theta)\ \frac{\hat{h}^3}{3}\ \hat{h}_\xi + \frac{\hat{h}^3}{3}\right]_\xi = 0, \tag{3.5}$$

where $\hat{D}(\theta) = \frac{D(\theta)}{Ca^{\frac{1}{3}}}$. Eq. (3.5) is subject to the boundary conditions,

$$\hat{h} \to 1,\ \hat{h}_{\xi\xi\xi} \to 0 \ \text{ as } \ \xi \to -\infty, \quad \text{and} \quad \hat{h} \to b,\ \ \hat{h}_{\xi\xi\xi} \to 0 \ \text{ as } \ \xi \to \infty. \tag{3.6}$$

Integrating Eq. (3.5) and using downstream boundary condition, we obtain

$$-c\hat{h} + \frac{\hat{h}^3}{3}\,\hat{h}_{\xi\xi\xi} - D(\theta)\,\frac{\hat{h}^3}{3}\,\hat{h}_{\xi} + \frac{\hat{h}^3}{3} = -\frac{1}{3}\left(b + b^2\right). \tag{3.7}$$

Applying the upstream boundary conditions, we obtain the wave speed $c = \frac{1}{3}(1+b+b^2)$. Hence, the travelling wave solution satisfies the following third order ODE,

$$\hat{h}_{\xi\xi\xi} = \frac{1+b+b^2}{\hat{h}^2} - \frac{b+b^2}{\hat{h}^3} + \hat{D}(\theta)\,\hat{h}_{\xi} - 1, \tag{3.8}$$

subject to the following boundary conditions,

$$\hat{h} \to 1 \;\text{ as }\; \xi \to -\infty, \quad \text{and} \quad \hat{h} \to b \;\text{ as }\; \xi \to \infty. \tag{3.9}$$

Eqs. (3.8,3.9) can be solved numerically either as an Initial-Value-Problem (IVP) using the shooting method or as Boundary-Value-Problem (BVP) using finite differences to discretise the spatial variables and solving the resulting system of nonlinear equations.

The shooting method

Here, we present a brief outline of the shooting method that is used to solve Eq. (3.8,3.9). Eq. (3.8) is linearised about the upstream boundary condition, i.e., $\hat{h} \to 1$ as $\xi \to -\infty$. This generates three initial conditions along with a shooting parameter that is utilized to integrate Eq. (3.8). This shooting parameter is varied so as to satisfy the

downstream boundary condition, i.e., $\hat{h} \to b$ as $\xi \to +\infty$. For a detailed description, the reader can refer to [79]. We define, $\hat{h}(\xi) = 1 + \delta\ \hat{H}(\xi)$, where $\delta \ll 1$. Substituting in Eq. (3.8) gives at $O(\delta)$,

$$\hat{H}_{\xi\xi\xi} = (b + b^2 - 2)\hat{H} + \hat{D}(\theta)\hat{H}_{\xi}. \tag{3.10}$$

The above ODE has characteristic equation with three roots, one of which is negative and the remaining two roots are positive. Here, the negative root is neglected to prevent the solution increasing exponentially as $\xi \to \infty$. We have three cases for the remaining two positive roots depending on inclination angle θ. If the roots are,

- Real and equal ($= \alpha$, say), then the solution is written as,

  $\hat{H}(\xi) = 1 + e^{\alpha\xi}\,(A + B\xi)\,, \quad \text{as} \quad \xi \to -\infty.$

- Real and distinct ($= \alpha$, β, say), then the solution is,

  $\hat{H}(\xi) = 1 + A\ e^{\alpha\xi} + B\ e^{\beta\xi}, \quad \text{as} \quad \xi \to -\infty.$

- Complex conjugate roots ($= \alpha \pm i\beta$, say, with $\alpha > 0$), the solution takes the form,

  $\hat{H}(\xi) = 1 + A\ e^{\alpha\xi} \cos\left(\beta\,(\xi - \xi_0)\right), \quad \text{as} \quad \xi \to -\infty.$

We only consider the case where the roots are complex conjugate. This is the case for large inclination angle θ. Using the translational invariance of Eq. (3.10), we choose ξ_0 to be large and negative such that $\cos\left(\beta\,(\xi - \xi_0)\right) = 1$. We write Eq. (3.10) as a system of three first order ODEs with initial conditions provided by the above linearisation about $h = 1$. We solve the resulting IVP using the MATLAB routine ODE15s [74] and

modify the shooting parameter A until the downstream boundary condition is satisfied.

Solving the BVP

We rewrite Eq. (3.8) as,

$$F(\hat{h}) = 0, \quad \text{where} \quad F(\hat{h}) = \hat{h}_{\xi\xi\xi} - \left[\frac{1+b+b^2}{\hat{h}^2} - \frac{b+b^2}{\hat{h}^3} + D(\theta)\ \hat{h}_\xi - 1\right]. \qquad (3.11)$$

We discretise the above equation using centred finite differences on either a uniform or non-uniform mesh resulting in a system of nonlinear algebraic equations. We use the MATLAB routine FSOLVE to solve this system which uses a quasi-Newton method [74]. It is crucial to have a good starting guess for the solution to converge; we use the corresponding solution obtained by the shooting method as a starting guess.

Figure 3.2 shows the numerical solution of Eq. (3.10) obtained using the shooting method (dashed blue line) with stepsize $\Delta\xi = 10^{-5}$ and the BVP method (solid red line) with mesh size $\Delta\xi = 5 \times 10^{-5}$. The parameter values are $Ca = 1$, $\theta = 90^o$ (so, $D = 0$) and $b = 10^{-2}$. The solution is translated so that the origin $\xi = 0$ is at the minimum value of h attained just before connecting onto the precursor film, the so-called effective contact line. The solution is characterised by a shock-like structure which develops just upstream of the contact line, the so-called capillary ridge, and a capillary wave that forms both upstream and downstream of the capillary ridge (the downstream capillary wave has a much smaller lengthscale and is not seen on the scale shown in Figure 3.2). The width of the capillary ridge region scales like $Ca^{1/3}$ (see Eq.

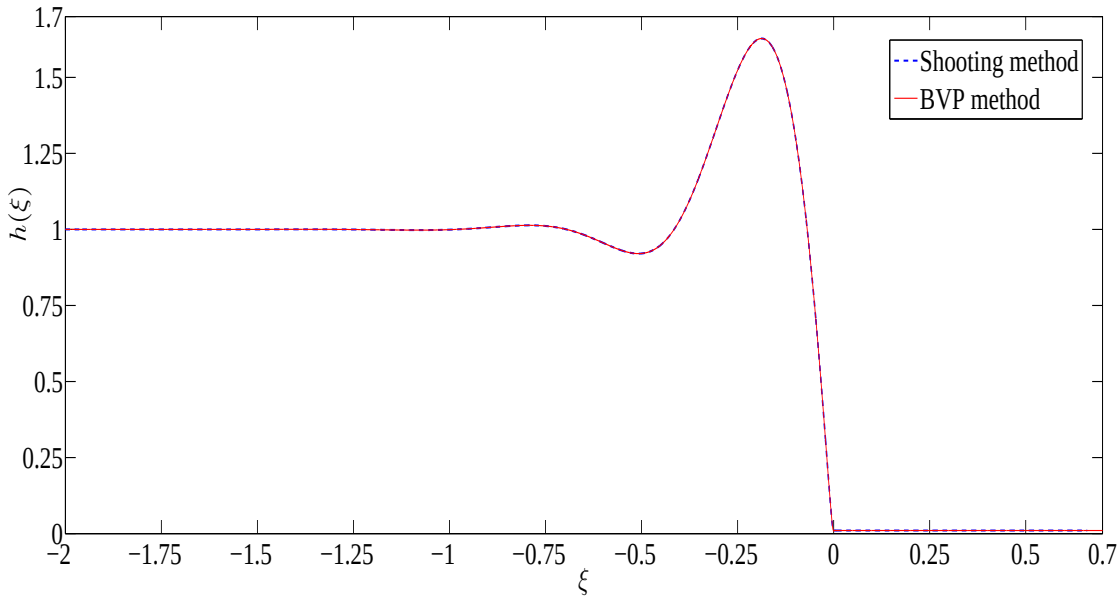

Figure 3.2: Numerical solution of Eq. (3.10) obtained using the shooting method (with stepsize $\Delta\xi = 10^{-5}$) and the BVP method (with mesh size $\Delta\xi = 5 \times 10^{-5}$). The parameter values are $Ca = 1$, $\theta = 90^o$ (so, $D = 0$) and $b = 10^{-2}$.

(3.4)) and hence becomes much steeper as $Ca \to 0$. This would require a large number of mesh points in any numerical discretisation to fully resolve this region making the use of an adaptive moving mesh even more important.

Figure 3.3(a) presents convergence histories (measured in the L_2 norm) for the shooting and the BVP methods. Both methods converge as the stepsize $\Delta\xi$ decreases. Figure 3.3(b) presents the error measured in the L_2 norm for both schemes. We assume the solution with $\Delta\xi = 10^{-5}$ obtained using shooting method as the exact solution when computing the error. We observe that the error in both methods tends to zero as $\Delta\xi$ goes to zero. Moreover, shooting method is more accurate than the BVP method. However, in the shooting method, the shooting parameter is dependent on the mesh size $\Delta\xi$. We can conclude that the shooting method is more accurate and convergent

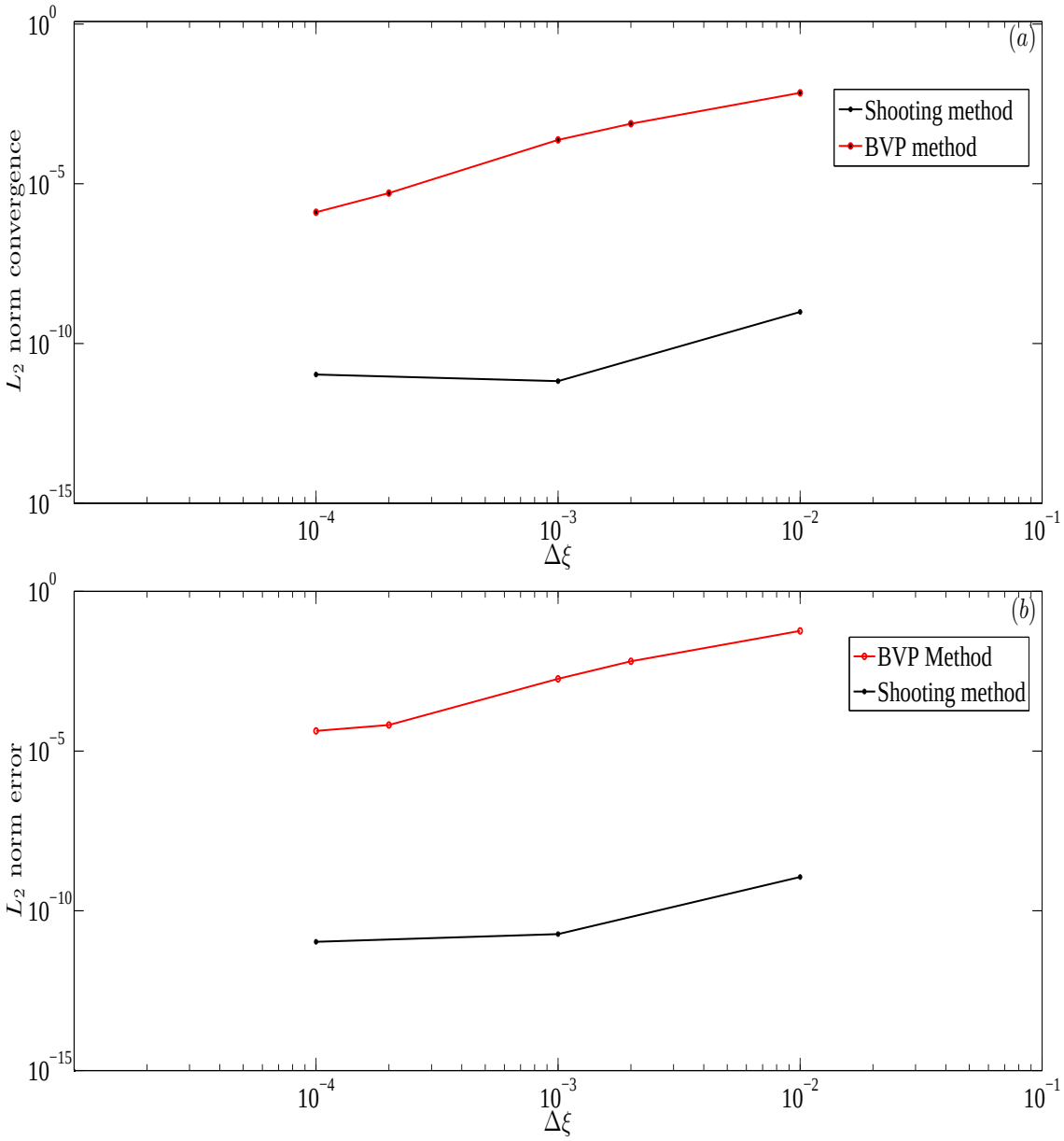

Figure 3.3: (a) convergence histories (measured using the L_2 norm) and (b) L_2 norm error for numerical solutions is obtained using the shooting and BVP methods for varying $\Delta\xi$. We assume that the travelling wave solution obtained using the shooting method for $\Delta\xi = 10^{-5}$ as the exact solution when comparing the error. The parameters values are $\theta = 90^o$ (so, $D = 0$) and $b = 10^{-2}$.

than the BVP method. We will assume that the numerical solution using the shooting method for $\Delta\xi = 10^{-5}$ as the exact solution when making comparisons between the numerical solution of the full time-dependent problem using a uniform and adaptive moving mesh scheme described in the following sections.

3.1.2 Numerical solution of Eqs. (3.1, 3.2) on a uniform mesh

The domain $[0, L]$ is uniformly divided into $N + 1$ discrete points as follows: $x_j = (j-1)\,\Delta x,\ \forall\ x_j \in [0, L],\quad j = 1, 2, ..., N+1$, where $\Delta x = \frac{L}{N}$ is the width of each sub-interval. We discretise the spatial derivatives appearing in Eq. (3.1) using finite differences; the time derivative is kept continuous.

The spatial discretisation is done as follows. We define a forward and a backward difference as: $h_{x,j+\frac{1}{2}} = \dfrac{h_{j+1} - h_j}{\Delta x} + O(\Delta x), \quad h_{\bar{x},j+\frac{1}{2}} = \dfrac{h_j - h_{j-1}}{\Delta x} + O(\Delta x)$, respectively. Thus, a semi-discretisation of Eq. (3.1), keeping the time derivative continuous, is

$$h_{t,j} + \frac{Ca}{3}\left[\frac{(h^3)_{j+\frac{1}{2}}\,(h_{x\bar{x}x})_{j+\frac{1}{2}} - (h^3)_{j-\frac{1}{2}}\,(h_{x\bar{x}x})_{j-\frac{1}{2}}}{\Delta x}\right] - \frac{D(\theta)}{3}\left[\frac{(h^3)_{j+\frac{1}{2}}\,(h_x)_{j+\frac{1}{2}} - (h^3)_{j-\frac{1}{2}}\,(h_x)_{j-\frac{1}{2}}}{\Delta x}\right] + \frac{1}{3}\left[\frac{(h^3)_j - (h^3)_{j-1}}{\Delta x}\right] = 0, \quad (3.12)$$

where $j = 2, ..., N$ and $(h^3)_{j+\frac{1}{2}}$ can be approximated by

$$(h^3)_{j+\frac{1}{2}} = \frac{h_{j+1}^3 + h_j^3}{2} \ \text{ or } \ (h^3)_{j+\frac{1}{2}} = \left(\frac{h_{j+1} + h_j}{2}\right)^3.$$

It has been shown that there are particular choices for the approximation of $(h^3)_{j+\frac{1}{2}}$ that have special properties, for example, the so-called positivity-preserving scheme, meaning that if one starts from strictly positive data for h, the scheme will help preserving this property. The interested reader is referred to the works by Bertozzi [6] and Zhornitskaya & Bertozzi [87] for details. Evaluating Eq. (3.12) at $j = 2, N$ require fictitious points which are obtained by discretising the boundary conditions $h_{xxx} = 0$ at $x = 0, L$ using centred finite differences. The boundary conditions $h(0, t) = 1$ and $h(L, t) = b$ are replaced by their ODE form:

$$h_{t,1} = 0, \quad h_{t,N+1} = 0. \tag{3.13}$$

This semi-discretisation scheme is second order accurate and uses a 5-point stencil. It requires a smaller bandwidth in comparison with a 7-point stencil if using a centred finite difference scheme. This discretisation is frequently used in thin film problems, particularly, for discretising the fourth order derivative (for example, see Kondic [57]).

Equations (3.12,3.13) form a system of $N + 1$ ordinary differential equations for the solution $h_1, \ldots, h_{N+1}$ with initial condition given by Eq. (3.3). These are solved using the stiff ODE solver DASSL [72] which uses backward differentiation formulae (BDFs) to approximate the time derivative and the resulting nonlinear system of equations are solved using Newton's method. The banded structure of the Jacobian matrix (with bandwitdh=5 based on the above semi-discretisation) corresponding to the linearised system is utilised to improve the performance of the ODE solver.

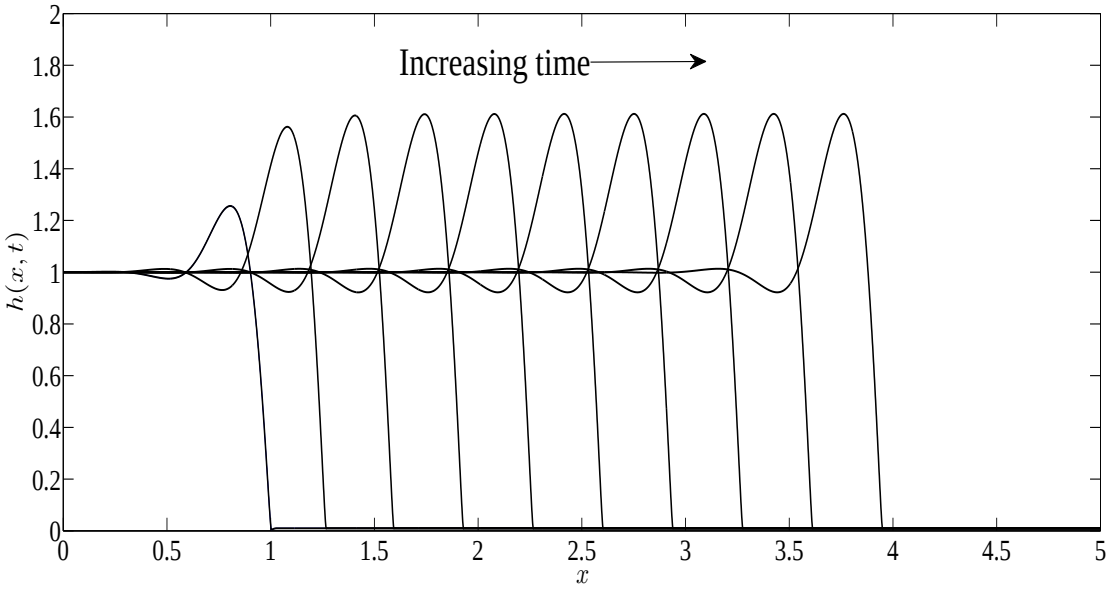

Figure 3.4: Time evolution of $h(x,t)$ to the travelling wave solution with parameter values $Ca = 10^{-3}$, $\theta = 90^o$, $b = 10^{-2}$, $L = 6$ and $\Delta x = 10^{-3}$ (so, $N = 6000$). Time t is increasing between $t = 0$ and $t = 10$.

Figure 3.4 illustrates the time evolution of h to a travelling wave solution with parameter values $Ca = 10^{-3}$, $\theta = 90^o$, $b = 10^{-2}$, $L = 6$ and $\Delta x = 10^{-3}$ (so, $N = 6000$). The results are shown for time t ranging between 0 to 10. Note that, after $t \approx 4$, the solution appears to have converged to a travelling wave solution. Figure 3.5(a) illustrates the numerical solution of Eq. (3.12) at $t = 10$ for varying Δx. The dashed line shows the assumed exact solution obtained using the shooting method with $\Delta x = 10^{-5}$. We note that the solution converges to the exact solution as Δx decreases. Therefore, we can conclude that the uniform scheme converges to the exact solution as $\Delta x \to 0$. This convergence is verified in figure 3.5(b) which shows the L_2 norm error between the numerical (at $t = 10$) and exact solution for decreasing Δx. Observe that L_2 norm error decreases as Δx decreases.

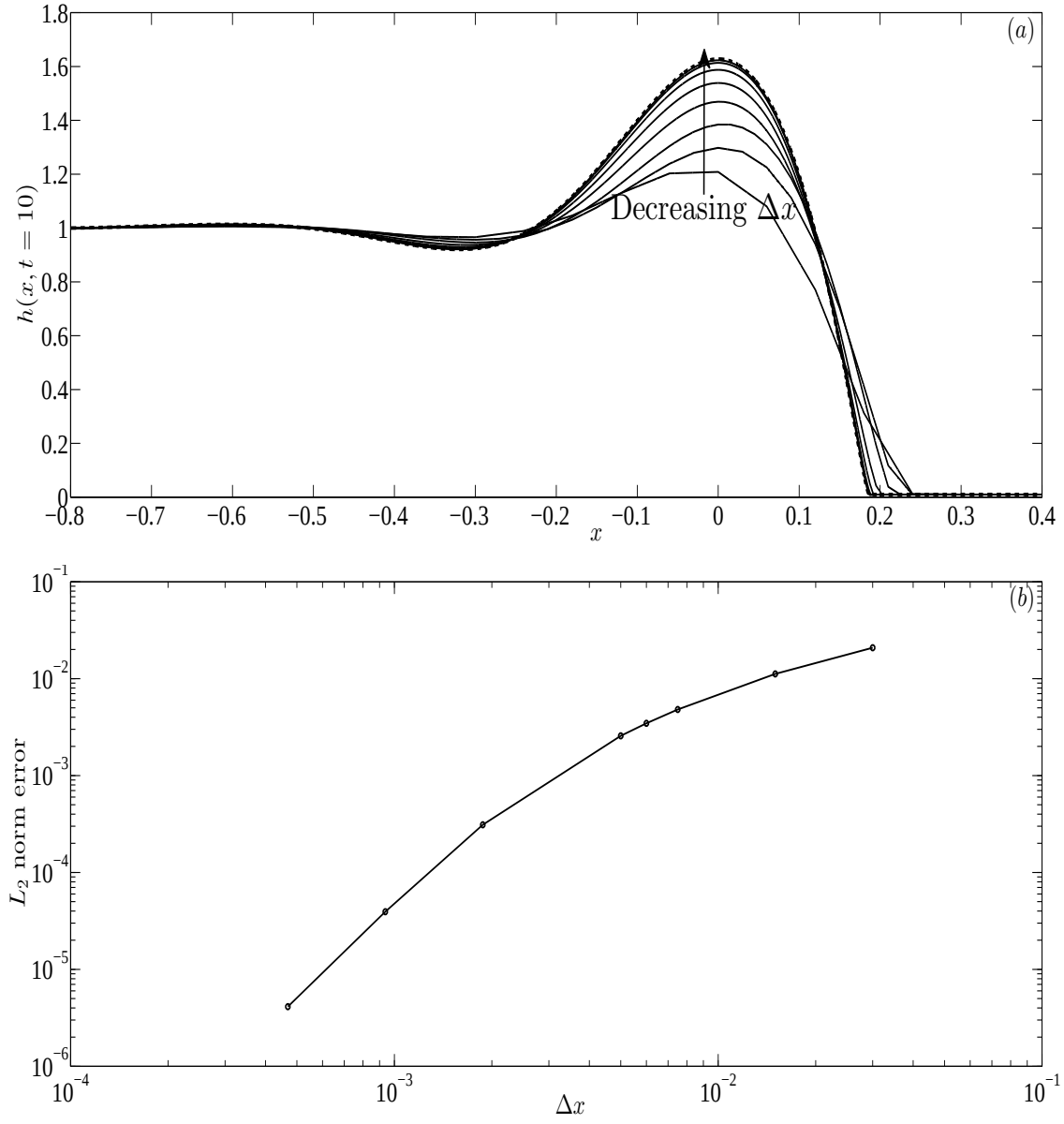

Figure 3.5: (a) shows the numerical solution of Eq. (3.12) at $t = 10$ for decreasing Δx. The dashed line shows the exact solution obtained using the shooting method with $\Delta x = 10^{-5}$, and (b) shows the L_2 norm error between the numerical (at $t = 10$) and exact solution for decreasing Δx.

3.1.3 Numerical solution of Eqs. (3.1, 3.2) on an adaptive moving mesh

We now seek the numerical solution of Eqs. (3.1, 3.2) using an adaptive moving mesh method. We first transform from the physical domain $\Omega_p \equiv [0, L]$ to the computational domain $\Omega_c \equiv [0, 1]$ using the transformation

$$x = x(\xi, t), \;\; \xi \in [0, 1], \;\; x \in [0, L].$$

Then the solution can be written as

$$\hat{h}(x, t) = h(x(\xi, t), t).$$

A moving mesh associated with the solution $\hat{h}$ is described as

$$\mathcal{J}_{\hat{h}}(t) \colon x_j(\xi) = x(\xi_j, t), \;\; j = 1, \ldots, N + 1, \tag{3.14}$$

and a fixed uniform mesh on the computational domain is described as

$$\mathcal{J}_h^c(t) \colon \xi_j = \frac{j - 1}{N}, \;\; j = 1, \ldots, N + 1. \tag{3.15}$$

Using the chain rule, we have

$$h_x \;=\; \frac{\hat{h}_\xi}{x_\xi}, \;\; h_t \;=\; \hat{h}_t \;-\; \frac{\hat{h}_\xi}{x_\xi}\, x_t. \tag{3.16}$$

Using the above, Eq. (3.1) can then be written as

$$\hat{h}_t - \frac{\hat{h}_\xi}{x_\xi} x_t = -\frac{\hat{Q}_\xi}{x_\xi} \;+ \frac{\hat{G}_\xi}{x_\xi}, \;\; \text{where} \tag{3.17}$$

$$\hat{Q} = Ca \left(\frac{\hat{h}^3}{3}\right) \frac{1}{x_\xi} \left(\frac{\left(\frac{\hat{h}_\xi}{x_\xi}\right)_\xi}{x_\xi} \right)_\xi, \;\; \hat{G} = D(\theta) \left(\frac{\hat{h}^3}{3}\right) \left(\frac{\hat{h}_\xi}{x_\xi}\right) - \left(\frac{\hat{h}^3}{3}\right). \tag{3.18}$$

A *conservative* semi-discretisation scheme for the spatial derivatives in Eq. (3.17) on the uniform mesh $\mathcal{J}_h^c(t)$ using finite differences can be written as, keeping the time derivative continuous,

$$\hat{h}_{t,j} - \frac{\hat{h}_{j+1} - \hat{h}_{j-1}}{x_{j+1} - x_{j-1}} x_{t,j} = -4\frac{\hat{Q}_j - \hat{Q}_{j-1}}{x_{j+2} - x_{j-2}} \;+ 2\frac{\hat{G}_j - \hat{G}_{j-1}}{x_{j+1} - x_{j-1}}, \;\; \text{where} \tag{3.19}$$

$$\hat{Q}_j = Ca \;\left(\frac{\hat{h}^3}{3}\right)_{j+1/2} \;\frac{1}{x_{\xi,j}} \left(\frac{\left(\frac{\hat{h}_\xi}{x_\xi}\right)_\xi}{x_\xi} \right)_{\xi,j}, \; G_j = \; D(\theta) \;\left(\frac{\hat{h}^3}{3}\right)_{j+1/2} \;\left(\frac{\hat{h}_\xi}{x_\xi}\right)_{,j} - \left(\frac{\hat{h}^3}{3}\right)_{j+1/2},$$

$j = 2, \ldots, N.$

The spatial derivatives appearing in Q_j are discretised as follows:

$$\frac{1}{x_{\xi,j}}\left(\frac{\left(\frac{\hat{h}_\xi}{x_\xi}\right)_\xi}{x_\xi}\right)_{\xi,j} = \frac{3}{x_{j+2}-x_{j-1}}\left(\frac{1}{x_{\xi,j+1}}\left(\frac{\hat{h}_\xi}{x_\xi}\right)_{\xi,j+1} - \frac{1}{x_{\xi,j}}\left(\frac{\hat{h}_\xi}{x_\xi}\right)_{\xi,j}\right), \quad \text{where}$$

$$\frac{1}{x_{\xi,j}}\left(\frac{\hat{h}_\xi}{x_\xi}\right)_{\xi,j} = \frac{2}{x_{j+1}-x_{j-1}}\left(\frac{\hat{h}_{j+1}-\hat{h}_j}{x_{j+1}-x_j} - \frac{\hat{h}_j-\hat{h}_{j-1}}{x_j-x_{j-1}}\right), \quad j=1,\ldots,N. \tag{3.20}$$

The spatial derivatives appearing in G_j are discretised as:

$$\left(\frac{\hat{h}_\xi}{x_\xi}\right)_{,j} = \frac{\hat{h}_{j+1}-\hat{h}_j}{x_{j+1}-x_j}. \tag{3.21}$$

In the above discretisations, $(\hat{h}^3)_{j+1/2}$ is approximated by $(\hat{h}^3)_{j+1/2} = \dfrac{\hat{h}_{j+1}^3+\hat{h}_j^3}{2}$ or $(\hat{h}^3)_{j+1/2} = \left(\dfrac{\hat{h}_{j+1}+\hat{h}_j}{2}\right)^3$. A similar approximation is used for evaluating $(\hat{h}^3)_{j-1/2}$. Evaluating Eq. (3.19) at $j = 2, N$ require fictitious points $\hat{h}_0$ and $\hat{h}_{N+2}$, which are obtained by discretising the boundary conditions $h_{xxx} = 0$ at $x = 0, L$ using centred finite differences.

The boundary conditions $h(0,t) = 1$ and $h(L,t) = b$ are replaced by their ODE form:

$$h_{t,1} = 0, \quad h_{t,N+1} = 0. \tag{3.22}$$

We can also write a *non-conservative* semi-discretisation scheme for the spatial deriva-

tives in Eq. (3.1). To do this, we write Eq. (3.1) in the following form:

$$h_t + Ca\left[\frac{h^3}{3}p_x\right]_x - D(\theta)\left[\frac{h^3}{3}h_x\right]_x + \left[\frac{h^3}{3}\right]_x = 0, \quad \text{where } p = h_{xx}. \tag{3.23}$$

A *non-conservative* semi-discretisation scheme for a function of the form $[au_x]_x$ on the uniform mesh $\mathcal{J}_h^c(t)$ using finite differences can be written as:

$$\begin{aligned}[au_x]_{x,j} = \frac{1}{x_{\xi,j}}\left[\frac{au_\xi}{x_\xi}\right]_{\xi,j} = \frac{1}{x_{\xi,j}}\left[\frac{[au_\xi]_{\xi,j}}{x_{\xi,j}} - \frac{1}{x_{\xi,j}^2}a_j u_{\xi,j}x_{\xi\xi,j}\right] = \\ \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2\left[a_{j+\frac{1}{2}}(u_{j+1}-u_j) - a_{j-\frac{1}{2}}(u_j-u_{j-1})\right] \\ - a_j\left(\frac{u_{j+1}-u_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right).\end{aligned} \tag{3.24}$$

Using the above, a *non-conservative* semi-discretisation scheme for the spatial derivatives in Eq. (3.23) on the uniform mesh $\mathcal{J}_h^c(t)$ using finite differences can be written as, keeping the time derivative continuous,

$$\hat{h}_{t,j} + Ca\left[\frac{\hat{h}^3}{3}p_x\right]_{x,j} - D(\theta)\left[\frac{\hat{h}^3}{3}\hat{h}_x\right]_{x,j} + \left[\frac{\hat{h}^3}{3}\right]_{x,j} = 0, \quad p_j = \hat{h}_{xx,j}, \tag{3.25}$$

where

$$[ap_x]_{x,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2 \left[a_{j+\frac{1}{2}}(p_{j+1}-p_j) - a_{j-\frac{1}{2}}(p_j - p_{j-1})\right]$$
$$- a_j \left(\frac{p_{j+1}-p_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right), \tag{3.26}$$

$$[a\hat{h}_x]_{x,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2 \left[a_{j+\frac{1}{2}}(\hat{h}_{j+1}-\hat{h}_j) - a_{j-\frac{1}{2}}(\hat{h}_j - \hat{h}_{j-1})\right]$$
$$- a_j \left(\frac{\hat{h}_{j+1}-\hat{h}_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right), \tag{3.27}$$

$$p_j = \hat{h}_{xx,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2 \left[\hat{h}_{j+1} - 2\hat{h}_j + \hat{h}_{j-1})\right]$$
$$- \left(\frac{\hat{h}_{j+1}-\hat{h}_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right), \tag{3.28}$$

$$[\hat{h}^3]_x = 2\left(\frac{\hat{h}^3_{j+1/2} - \hat{h}^3_{j-1/2}}{x_{j+1}-x_{j-1}}\right). \tag{3.29}$$

Here, $a = \frac{1}{3}\hat{h}^3$ and approximations to $a_{j+1/2}$ and $a_{j-1/2}$ are the same as defined previously.

The equidistributing coordinate transformation $x = x(\xi, t)$ is obtained by solving the moving mesh PDEs (MMPDEs). We use four different MMPDEs, the so-called MMPDE 4, 5 and 6 and modified MMPDE 5, described previously in Chapter 2. A semi-discretisation scheme using centred finite differences to discretise the spatial derivatives

is employed. The semi-discretisation of the above MMPDEs is as follows:

$$\begin{aligned}
\text{MMPDE 4}: \quad & (\hat{\rho}x_{t\xi})_\xi = -\frac{1}{\tau}(\hat{\rho}x_\xi)_\xi, \\
\text{semi-discretisation}: \quad & \tau[(\hat{\rho}_{j+1}+\hat{\rho}_j)(x_{t,j+1}-x_{t,j}) - (\hat{\rho}_{j-1}+\hat{\rho}_j)(x_{t,j}-x_{t,j-1})] = \\
& -[(\hat{\rho}_{j+1}+\hat{\rho}_j)(x_{j+1}-x_j) - (\hat{\rho}_{j-1}+\hat{\rho}_j)(x_j-x_{j-1})], \\
& j = 2,\ldots,N. \qquad (3.30)
\end{aligned}$$

$$\begin{aligned}
\text{MMPDE5}: \quad & x_t = \frac{1}{\tau}(\hat{\rho}x_\xi)_\xi, \\
\text{semi-discretisation}: \quad & \tau x_{t,j} = \frac{1}{2\Delta\xi^2}[(\hat{\rho}_{j+1}+\hat{\rho}_j)(x_{j+1}-x_j) - (\hat{\rho}_{j-1}+\hat{\rho}_j)(x_j-x_{j-1})], \\
& j = 2,\ldots,N. \qquad (3.31)
\end{aligned}$$

$$\begin{aligned}
\text{MMPDE6}: \quad & x_{t,\xi\xi} = -\frac{1}{\tau}(\hat{\rho}x_\xi)_\xi, \\
\text{semi-discretisation}: \quad & \tau(x_{t,j+1}-2x_{t,j}+x_{t,j-1}) = -\frac{1}{2}[(\hat{\rho}_{j+1}+\hat{\rho}_j)(x_{j+1}-x_j) - \\
& (\hat{\rho}_{j-1}+\hat{\rho}_j)(x_j-x_{j-1})], \;\; j = 2,\ldots,N. \qquad (3.32)
\end{aligned}$$

$$\begin{aligned}
\text{Modified MMPDE5}: \quad & x_t = \frac{1}{\hat{\rho}\tau}(\hat{\rho}x_\xi)_\xi, \\
\text{semi-discretisation}: \quad & \tau x_{t,j} = \frac{1}{2\hat{\rho}_j\Delta\xi^2}[(\hat{\rho}_{j+1}+\hat{\rho}_j)(x_{j+1}-x_j) - \\
& (\hat{\rho}_{j-1}+\hat{\rho}_j)(x_j-x_{j-1})], \;\; j = 2,\ldots,N. \qquad (3.33)
\end{aligned}$$

Here, $\hat{\rho}(x,t)$ is a monitor function (defined below), $\Delta\xi = \frac{1}{N}$ is the mesh size in the computational domain and $\tau > 0$ is a user-specified relaxation parameter. τ is the timescale over which the mesh responds to changes in the monitor function $\hat{\rho}(x,t)$. The smaller τ, the more quickly the mesh responds to changes in $\hat{\rho}(x,t)$. Likewise, the mesh moves slowly when a large value of τ is used. MMPDE 5 and modified MMPDE 5 and their discretisation given in Eqs. (3.31,3.33) are generally quite stiff and in practice it is proposed to use a more regular system,

$$\begin{aligned}
\text{Regularised MMPDE5}: \quad & x_t - \gamma_1 x_{t,\xi\xi} = \frac{1}{\tau}(\hat{\rho}x_\xi)_\xi, \\
\text{semi-discretisation}: \quad & \tau[x_{t,j} - \frac{\gamma_1}{2\Delta\xi^2}(x_{t,j+1} - 2x_{t,j} + x_{t,j-1})] = \\
& \frac{1}{2\Delta\xi^2}[(\hat{\rho}_{j+1} + \hat{\rho}_j)(x_{j+1} - x_j) - (\hat{\rho}_{j-1} + \hat{\rho}_j)(x_j - x_{j-1})], \\
& j = 2, \ldots, N. \qquad (3.34)
\end{aligned}$$

$$\begin{aligned}
\text{Regularised modified MMPDE5}: \quad & x_t - \gamma_1 x_{t,\xi\xi} = \frac{1}{\hat{\rho}\tau}(\hat{\rho}x_\xi)_\xi, \\
\text{semi-discretisation}: \quad & \tau[x_{t,j} - \frac{\gamma_1}{2\Delta\xi^2}(x_{t,j+1} - 2x_{t,j} + x_{t,j-1})] = \\
& \frac{1}{2\hat{\rho}_j\Delta\xi^2}[(\hat{\rho}_{j+1} + \hat{\rho}_j)(x_{j+1} - x_j) - (\hat{\rho}_{j-1} + \hat{\rho}_j)(x_j - x_{j-1})], \\
& j = 2, \ldots, N. \qquad (3.35)
\end{aligned}$$

Here, the parameter $\gamma_1 > 0$ is related to the monitor function $\hat{\rho}$ (see Budd *et al.* [13] and references therein).

The boundary conditions $x(0,t) = 0$ and $x(1,t) = L$ are replaced by their ODE form:

$$x_{t,1} = 0, \;\; x_{t,N+1} = 0. \tag{3.36}$$

The initial condition

$$x(\xi, 0) = L\xi, \tag{3.37}$$

which represents a uniform initial mesh on the physical domain $\Omega_p \equiv [0, L]$. We note here that a non-uniform initial mesh was also used which was obtained by solving in pseudo-time the chosen MMPDE (with the uniform mesh as the initial condition) with $\hat{h}$ fixed (hence, $\hat{\rho}$ is also fixed) at its initial condition given by Eq. (3.3). This did not have any significant influence on the solution or the performance of the solver in comparison to the uniform initial mesh.

Finally, the monitor function $\hat{\rho}(x,t)$ is chosen and discretised using finite differences as:

$$\begin{aligned}
&\text{Arc-length monitor function :} \quad \hat{\rho}(x,t) = \sqrt{1 + |\hat{h}_x|^2}, \\
&\text{semi-discretisation :} \quad \hat{\rho}(x_j,t) = \sqrt{1 + |\hat{h}_{x,j}|^2}, \\
&\hat{h}_{x,j} := \begin{cases} \frac{\hat{h}_{j+1} - \hat{h}_{j-1}}{x_{j+1} - x_{j-1}}, & j = 2, \ldots, N \\ \frac{\hat{h}_2 - \hat{h}_1}{x_2 - x_1}, & j = 1 \\ \frac{\hat{h}_{N+1} - \hat{h}_N}{x_{N+1} - x_N}, & j = N+1. \end{cases}
\end{aligned} \tag{3.38}$$

Curvature-based monitor function : $\hat{\rho}(x,t) = (1+\alpha|\hat{h}_{xx}|^2)^{\frac{1}{4}}$,

semi-discretisation : $\hat{\rho}(x_j,t) = (1+\alpha|\hat{h}_{xx,j}|^2)^{\frac{1}{4}}$,

$$\hat{h}_{xx,j} := \begin{cases} \frac{2}{(x_{j+1}-x_{j-1})}\left[\frac{(\hat{h}_{j+1}-\hat{h}_j)}{(x_{j+1}-x_j)} - \frac{(\hat{h}_j-\hat{h}_{j-1})}{(x_j-x_{j-1})}\right], & j=2,...,N, \\ \frac{2\left[(x_2-x_1)(\hat{h}_3-\hat{h}_1)-(x_3-x_1)(\hat{h}_2-\hat{h}_1)\right]}{(x_3-x_1)(x_3-x_2)(x_2-x_1)}, & j=1, \\ \frac{2\left[(x_N-x_{N+1})(\hat{h}_{N-1}-\hat{h}_{N+1})-(x_{N-1}-x_{N+1})(\hat{h}_N-\hat{h}_{N+1})\right]}{(x_{N-1}-x_{N+1})(x_N-x_{N+1})(x_{N-1}-x_N)}, & j=N+1. \end{cases} \tag{3.39}$$

Here, α is user-defined parameters.

If $\hat{h}$ is not smooth, the discretised monitor function computed as above can change abruptly and slow down the computation. To obtain a smoother mesh and also make the MMPDEs easier to integrate, it is common practice in the context of moving mesh methods to smooth the monitor function. A simple but effective smoothing scheme suggested by Huang [53, 49] is based on weighted averaging,

$$\hat{\rho}_j := \sqrt{\frac{\sum_{k=j-p}^{j+p} \hat{\rho}_k^2 \left(\frac{\gamma}{1+\gamma}\right)^{|k-j|}}{\sum_{k=j-p}^{j+p} \left(\frac{\gamma}{1+\gamma}\right)^{|k-j|}}}, \quad j=1,\ldots,N+1 \tag{3.40}$$

where p is a non-negative integer called the smoothing index and γ is a positive smoothing parameter. Several sweeps of the scheme may be applied each integration step (only one sweep is used for the numerical results presented in this chapter).

The above semi-discretisations for $\hat{h}$ and x form a coupled system of $2(N+1)$ ordinary differential equations for the solution $\hat{h}_1,\ldots,\hat{h}_{N+1}$ and the mesh $x_1,\ldots,x_{N+1}$ with initial condition given by Eqs. (3.3,3.37). These are solved using the stiff ODE

solver DASSL [72]. We use a staggered system for numbering the unknowns, i.e., $\hat{h}_1, x_1, \hat{h}_2, x_2, \ldots, \hat{h}_{N+1}, x_{N+1}$ which provides a smaller bandwidth for the Jacobian matrix (note: the bandwidth=10 for the semi-discretisation in Eq. (3.19); it could be larger if smoothing of monitor function is used. This is in comparison to, for example, the numbering $\hat{h}_1, \hat{h}_2, \ldots, \hat{h}_{N+1}, x_1, x_2, \ldots, x_{N+1}$, which although sparse has a much bigger bandwidth. This significantly improves the performance of the solver. We note here that the stiff solver DASPK [12] was also used which utilises an iterative scheme based on Krylov subspace methods for the solution of the linearised system (DASSL uses a direct solver) including preconditioning using Incomplete LU decomposition of the Jacobian matrix. This had no significant influence on the performance in comparison to DASSL, at least for the 1D problem considered in this chapter. We do not show here but the numerical results show that

- If the value of the capillary number Ca decreases, the width of the capillary ridge decreases.
- If the inclination angle θ increases, the height of the capillary ridge increases.
- If the value of the precursor film thickness b decreases, the height of the capillary ridge increases.

In all the results presented below, the parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-2}$ and $L = 6$; the monitor function smoothing parameters are $p = 2$, $\gamma = 2$. The choice of these parameters particularly $Ca, b \ll 1$, other than being realistic is that it makes the problem stiff and challenging to resolve numerically

the internal layers associated with the capillary ridge region. We only show results based on the conservative semi-discretisation scheme Eq. (3.20). Our results using the non-conservative scheme Eq. (3.25) was not as accurate compared to the conservative scheme so we do not show these results here. The first set of results show the evolution of $h(x,t)$ and $x(\xi,t)$ with MMPDE and monitor function chosen to be MMPDE 2 (with relaxation parameter $\tau = 10^{-2}$) and curvature-based monitor function (with $\alpha = 1$), respectively. We then consider the error and convergence of the moving adaptive mesh scheme for varying MMPDEs (and their parameters) and monitor function.

Figure 3.6(a,b) illustrate the time evolution of $h(x,t)$ to the travelling wave solution and the corresponding trajectories $x(\xi,t)$, respectively, obtained using the adaptive moving mesh scheme with $N = 800$ (so, initial $\Delta x = 0.0075$). In this simulation, MMPDE 2 (using $\tau = 10^{-2}$) with the curvature-based monitor function (using $\alpha = 1$) are used. The results are shown for t ranging between 0 to 10. Figure 3.6(a) shows that the adaptive moving mesh scheme captures the essential features of the solution, including the capillary ridge and capillary wave ahead and behind it, with less number of points ($N = 800$ here compared to $N = 6000$ for the uniform mesh computation shown in figure 3.4). Figure 3.6(b) shows that the mesh points are redistributed with more points in the region near the capillary ridge and fewer points elsewhere ($\Delta x \approx 10^{-4}$ in the capillary ridge region and $\Delta x \approx 10^{-2}$ elsewhere; the initial $\Delta x = 0.0075$ uniform everywhere).

Figure 3.7(a) presents the solution $h(x,t)$ at time $t = 10$. The insets show the in-

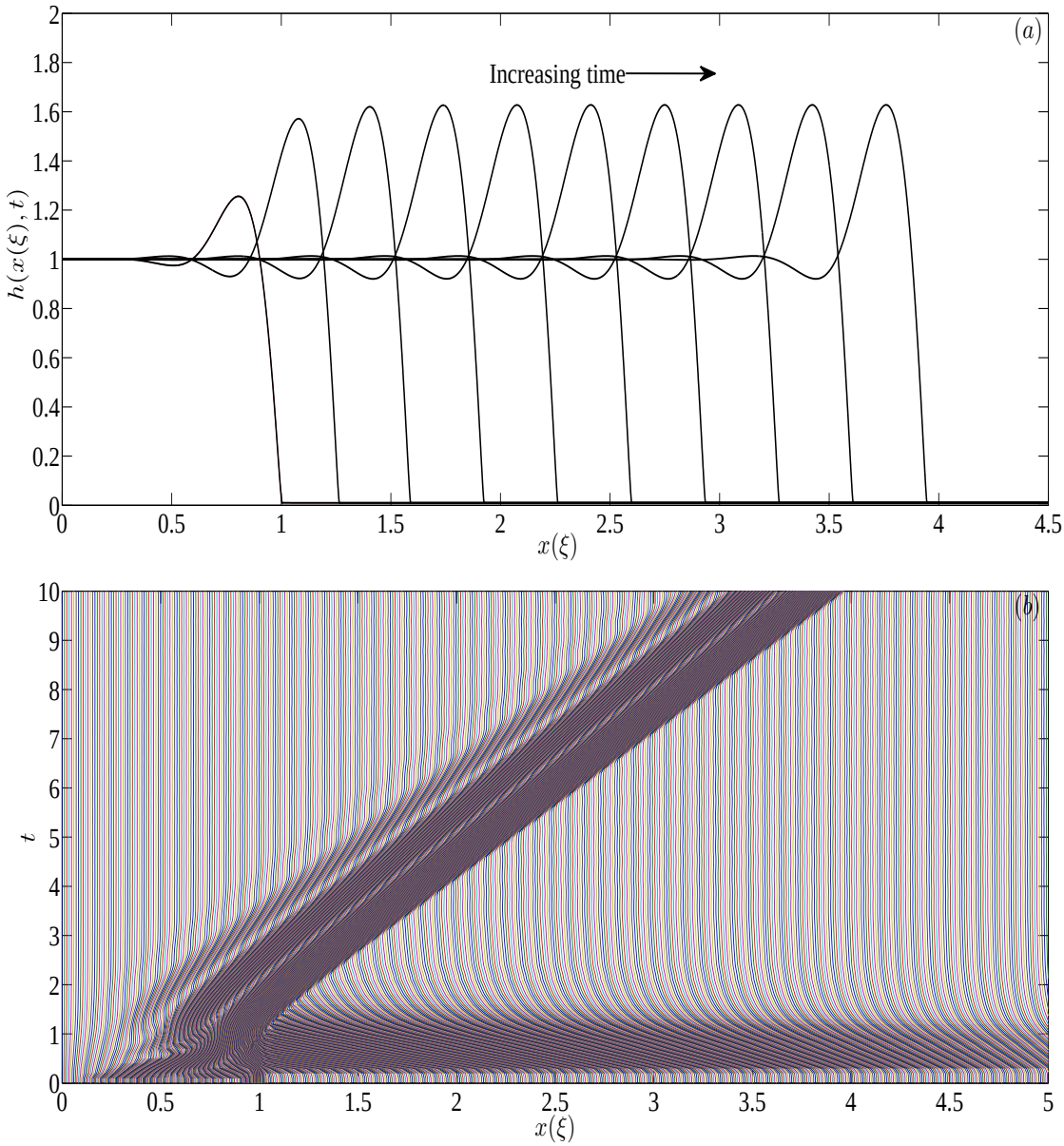

Figure 3.6: (a) time evolution of $h(x,t)$ to the travelling wave solution and (b) the corresponding mesh trajectories $x(\xi,t)$ obtained using the moving adaptive scheme with $N = 800$ (initial $\Delta x = 0.0075$), MMPDE2 and curvature monitor function. The time t ranges between 0 to 10. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-2}$, $L = 6$, $p = 2$, $\gamma = 2$, $\alpha = 1$ and $\tau = 10^{-2}$.

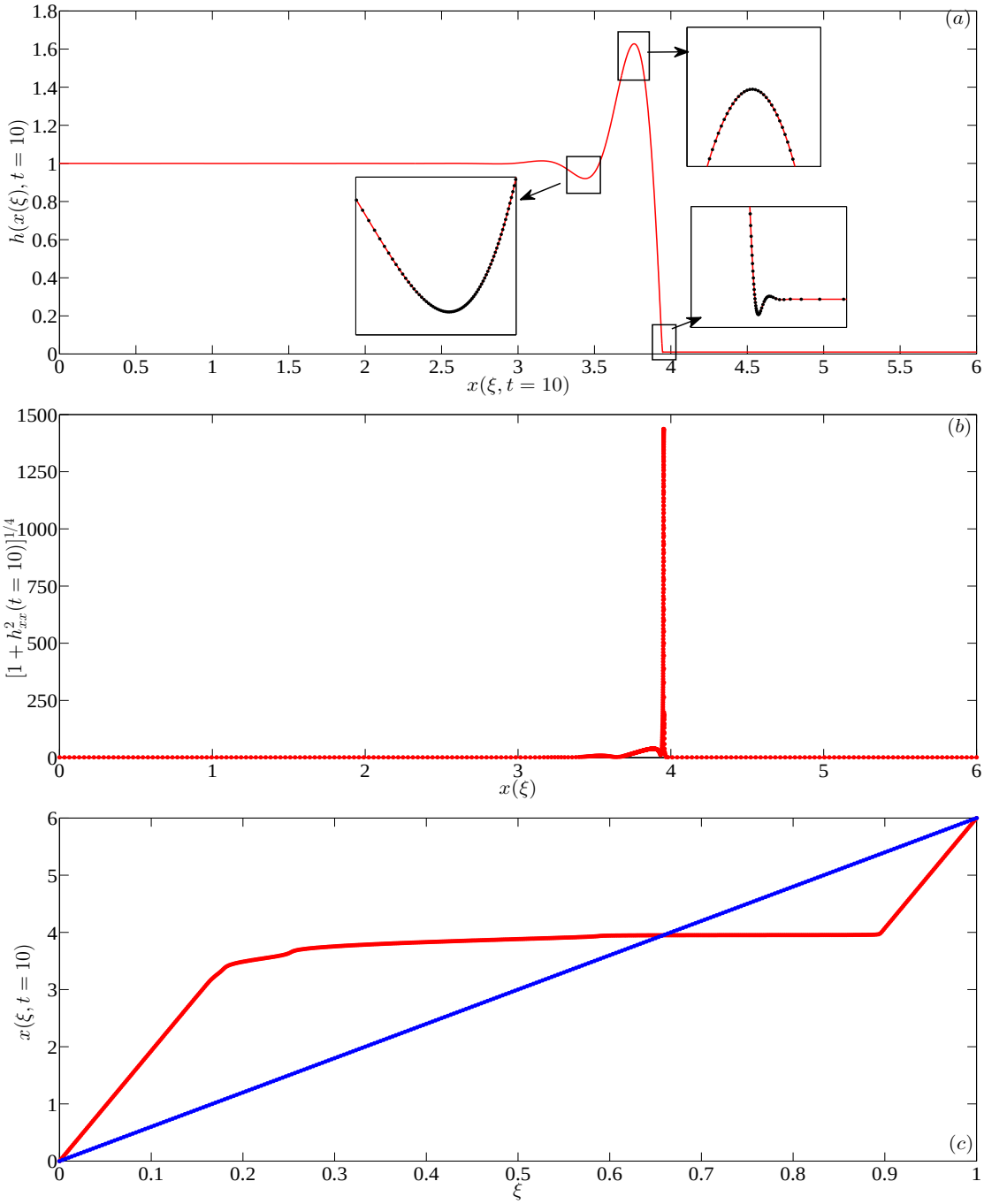

Figure 3.7: (a) $h(x, t = 10)$, (b) curvature-based monitor function at $t = 10$ and (c) $x(\xi, t = 10$ obtained using the moving adaptive scheme with $N = 800$ (initial $\Delta x = 0.0075$) and MMPDE2. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-2}$, $L = 6$, $p = 2$, $\gamma = 2$, $\alpha = 1$ and $\tau = 10^{-2}$. Insets in (a) show the zoomed-in capillary ridge and capillary wave ahead and behind the ridge. The solid blue line in (c) represents a uniform mesh.

creased number of points that are redistributed to the capillary ridge and the capillary wave ahead and behind it. Figure 3.7(b) shows the curvature-based monitor function associated with this solution. The large values of the curvature in the capillary ridge region results in the increased number of points redistributed in this region (see insets in figure 3.7(a)). Figure 3.7(c) shows the equidistributing coordinate transformation $x = x(\xi, t = 10)$. We observe the large number of points in the capillary ridge region (around $x = 4$) compared to elsewhere (compare also to a uniform mesh represented by the solid blue line). Hence, the adaptive moving mesh scheme allocates large number of points where there is a rapid variation in the solution characterized by large variations in its curvature.

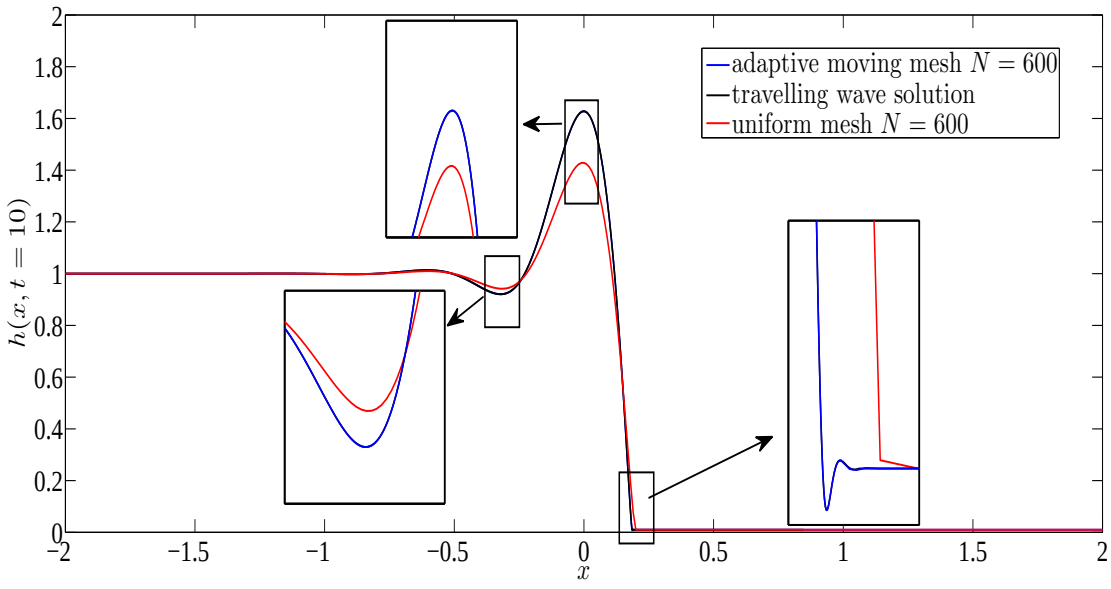

Figure 3.8: This figure presents the travelling wave solution (solid black line) and corresponding numerical results for both uniform (solid red line) and adaptive moving mesh (solid blue line)schemes. Insets show the zoomed-in solution near the capillary ridge and capillary wave ahead and behind the ridge. The adaptive moving mesh solution is obtained using MMPDE 2 and the curvature-based monitor function. The parameter values are: $N = 600$ (initial $\Delta x = 10^{-2}$), $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-2}$, $L = 6$, $p = 2$, $\gamma = 2$, $\alpha = 1$, and $\tau = 10^{-2}$.

Figure 3.8 compares the assumed exact travelling wave solution (solid black line) with that computed on a uniform mesh (solid red line) and adaptive moving mesh (solid blue line). To make a direct comparison between the uniform and adaptive moving mesh schemes, we fix $N = 600$ points for both schemes (initial $\Delta x = 10^{-2}$). We illustrate the accuracy of the numerical solution by focussing on the capillary ridge and the capillary wave ahead and behind it (see insets in figure 3.8). We observe that the solution obtained using the moving adaptive mesh scheme is almost identical to the assumed exact solution (see solid blue line in the insets in figure 3.8). The numerical solution using the uniform mesh scheme has not converged to the exact solution for the value of $\Delta x = 10^{-2}$ used (see solid red line in the insets in figure 3.8). The adaptive moving mesh scheme takes an average value of $\Delta x = 2 \times 10^{-5}$ in the capillary ridge and capillary wave regions, hence the almost identical match to the exact solution. However, the uniform mesh scheme requires at least $\Delta x \leq 10^{-3}$ ($N = 6000$ points for this case) for this region to be well-resolved (not shown here). The adaptive moving mesh scheme requires much fewer points ($N = 600$ or initial $\Delta x = 10^{-2}$) to give much higher resolution and accuracy of the numerical solution.

Error analysis and convergence

We now consider the error and convergence of the moving adaptive mesh scheme for varying MMPDEs (and their parameters) and monitor function. Table 3.1 summarises the error and CPU time taken to reach $t = 10$ using MMPDEs $4, 5, 6$ and modified

MMPDE	N	τ	CPU	Error
4	1000	1	41s	3.8×10^{-3}
	1000	10^{-1}	64s	1.8×10^{-4}
	1000	10^{-2}	71s	3.2×10^{-9}
	1000	10^{-3}	148s	1.0×10^{-8}
5	1000	1	48s	2.4×10^{-3}
	1000	10^{-1}	14650s	5.9×10^{-8}
6	1000	1	79s	1×10^{-5}
	1000	10^{-1}	80s	2.2×10^{-8}
	1000	10^{-2}	130s	1.1×10^{-8}
	1000	10^{-3}	932s	1.1×10^{-8}
modified 5	1000	1	116s	2.6×10^{-5}
	1000	10^{-1}	2088s	1.4×10^{-6}

Table 3.1: Error and CPU time taken to reach $t = 10$ for MMPDEs 4, 5 and 6, and modified MMPDE 5 varying the relaxation parameter τ. The numerical solution for h used is obtained at $t = 10$ and the curvature-based monitor function is used. The parameter values are: $N = 1000$ (initial $\Delta x = 6 \times 10^{-3}$), $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-2}$, $L = 6$, $p = 2$, $\gamma = 2$, $\alpha = 1$.

MMPDE 5 for several values of τ with $N = 1000$ (initial $\Delta x = 6 \times 10^{-3}$). The curvature monitior function is used. The numerical solution for h used in calculating the error and CPU time is obtained at $t = 10$. We observe from Table 3.1 that MMPDE4 and MMPDE6 give more accurate solution and take less CPU time compared to the MMPDE 5 and modified MMPDE 5. Four values of the relaxation parameter τ are used with MMPDE4 and MMPDE 6 and we observe that the error gets smaller but the CPU time increases as τ decreases. The equations become much stiffer as τ decreases and we were unable to obtain a numerical solution for $\tau < 10^{-4}$ for MMPDE 4 and MMPDE 6. The same trend holds for MMPDE 5 and modified MMPDE 5, however, we were unable to obtain a numerical solution for $\tau < 10^{-2}$. Therefore, it can be concluded that a value of $\tau = 10^{-3} - 10^{-2}$ is optimal with respect to accuracy and

CPU time taken. Also, MMPDE 4 or MMPDE6 are the best in terms of accuracy and CPU time taken (see Table 3.1).

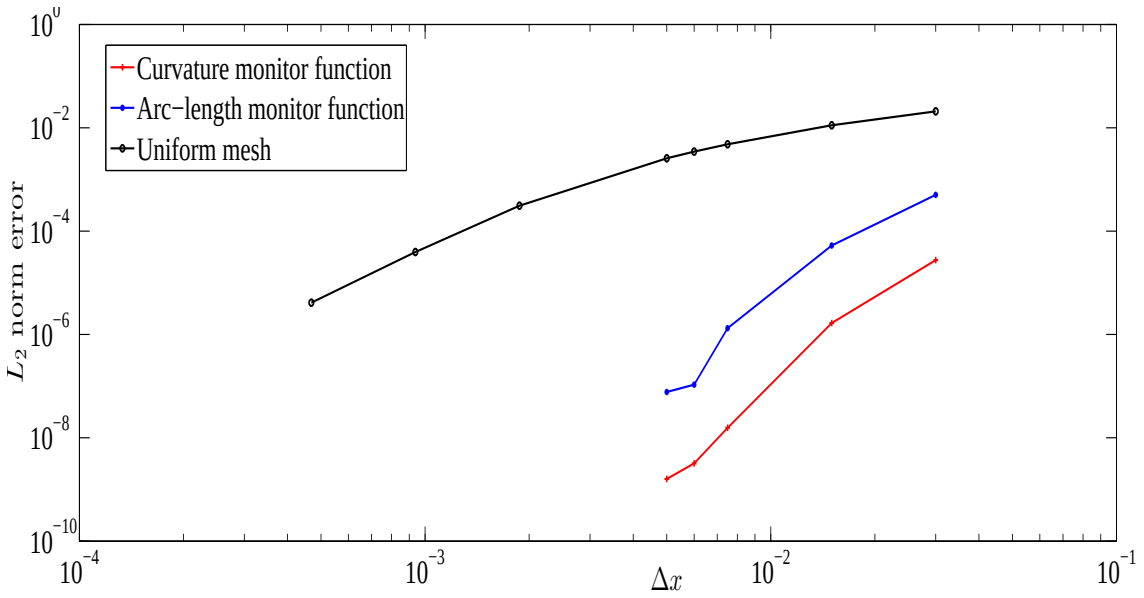

Figure 3.9: The L_2 norm error for numerical solutions obtained using uniform mesh (solid black line) and adaptive moving mesh using the arc-length monitor function (solid blue line) and curvature-based monitor function (solid red line). The numerical solution is obtained at $t = 10$ using MMPDE 4. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-2}$, $L = 6$, $p = 2$, $\gamma = 2$, $\alpha = 1$ and $\tau = 10^{-2}$.

Figure 3.9 compares the error measured in the L_2 norm for the numerical solutions obtained using the adaptive moving mesh (solid blue line using the arc-length monitor function and solid red line using the curvature-based monitor function) and uniform mesh schemes for varying Δx. The numerical solution for both schemes used in measuring the error is obtained at $t = 10$ and MMPDE 4 is used to adapt and move the mesh. We observe that the adaptive moving mesh numerical solution is more accurate with respect to the error compared to the uniform mesh scheme, i.e., it achieves a higher accuracy for the same number of mesh points. For example, the lowest error

recorded is 1.6×10^{-9} for $\Delta x = 5 \times 10^{-3}$ (corresponding to $N = 1200$ points) for the adaptive moving mesh solution using the curvature-based monitor function (solid red line). The error for the uniform mesh solution corresponding to this Δx is 2.57×10^{-3}. Moreover, it is also efficient in the number of mesh points used to achieve a desired level of accuracy, i.e., it uses less number of points to achieve the same error as the uniform mesh scheme. For example, the lowest value of the error recorded for the uniform mesh solution is 4.12×10^{-6} for $\Delta x = 5 \times 10^{-4}$ (corresponding to $N = 12000$ points). The adaptive moving mesh requires $\Delta x \approx 0.02$ (corresponding $N = 300$ points) to record similar error. We also observe that the adaptive moving mesh scheme using the curvature-based monitor function (solid red line) is more accurate and efficient in the number of points used compared to that using the arc-length monitor function (solid blue line). Therefore, it can be concluded that the adaptive moving mesh scheme using the curvature-based monitor function is optimal for the problem considered here.

N	Error		CPU time taken to $t = 10$	
	Uniform mesh	Adaptive moving mesh	Uniform mesh	Adaptive moving mesh
200	2.08×10^{-2}	2.75×10^{-5}	2s	9s
400	1.11×10^{-2}	1.66×10^{-6}	6.5s	29s
800	4.8×10^{-3}	1.56×10^{-8}	22s	59s
1000	3.4×10^{-3}	3.2×10^{-9}	25s	71s
1200	2.57×10^{-3}	1.6×10^{-9}	43s	119s
3000	3.1×10^{-4}	–	233s	–
6000	3.93×10^{-5}	–	264s	–
12000	4.12×10^{-6}	–	819s	–

Table 3.2: Comparing the error and CPU time taken to reach $t = 10$ for the uniform mesh and adaptive moving mesh (using the curvature-based monitor function and MMPDE 4) schemes. The numerical solution is obtained at $t = 10$. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-2}$, $L = 6$, $p = 2$, $\gamma = 2$, $\alpha = 1$ and $\tau = 10^{-2}$.

Table 3.2 shows the error and CPU time for the uniform mesh and adaptive moving mesh (using the curvature-based monitor function and MMPDE 4) schemes. The numerical solution used in recording the error is obtained at $t = 10$. The error columns summarise figure 3.9 (solid black line for uniform mesh scheme and solid red line for adaptive moving mesh scheme using the curvature-based monitor function and MMPDE 4). The error for the adaptive moving mesh scheme is much smaller and is achieved using less number of points compared to the uniform mesh scheme. However, with respect to the CPU time taken, the adaptive moving mesh takes more time to reach $t = 10$ compared to the uniform mesh scheme for the same number of points. This is due to the additional adaptive mesh equations that need to be simultaneously solved along with the discretised PDE. Hence, one would need to balance the accuracy desired and the CPU time taken to judge the efficacy of the moving adaptive mesh scheme over the uniform mesh scheme. For example, at $N = 800$ there is a threefold increase in time taken by the adaptive moving mesh scheme to reach $t = 10$ (less than a minute, though), however, there is a reduction in the error by five orders of magnitude. For this case, we can certainly conclude that the adaptive moving mesh scheme is more computationally efficient than the uniform mesh scheme. Note that we were unable to compute the solution at $t = 10$ for $N > 1200$. For these values of N the minimum Δx becomes very small and round-off errors dominate resulting in the numerical solution losing stability and becoming unstable.

Informed by the results of this section, we now consider the numerical solution of the

same PDE as in Eq. (3.1) but with constant volume boundary conditions.

3.2 PDE with constant volume boundary conditions

In this section, we investigate the numerical solution of a spreading one-dimensional drop using both a uniform mesh as well as an adaptive moving mesh. The underlying PDE is given by Eq. (3.1). This PDE is supplemented by boundary conditions, which are,

$$h = b, h_x, h_{xxx}, = 0 \text{ at } x = \pm L, \tag{3.41}$$

where $\pm L$ are the boundaries of the physical domain and $b \ll 1$, is the precursor film thickness. This represents no flux of fluid out of the domain and the drop connects onto a flat precursor film. The initial condition is chosen as:

$$h(x,0) = (1-x^2)\left[H(x_0-x) - H(-x_0-x)\right] + b\left[H(x-x_0) + H(-x_0-x)\right], \tag{3.42}$$

where $H(x)$ is the Heaviside function and x_0, is the initial location where the parabolic-shaped drop connects to the precursor film both upstream and downstream. This initial condition is shown in figure 3.10.

We seek the numerical solution of Eqs. (3.1) subject to the BCs given by Eq. (3.41) for the time evolution of the initial drop. We first show this solution using a uniform

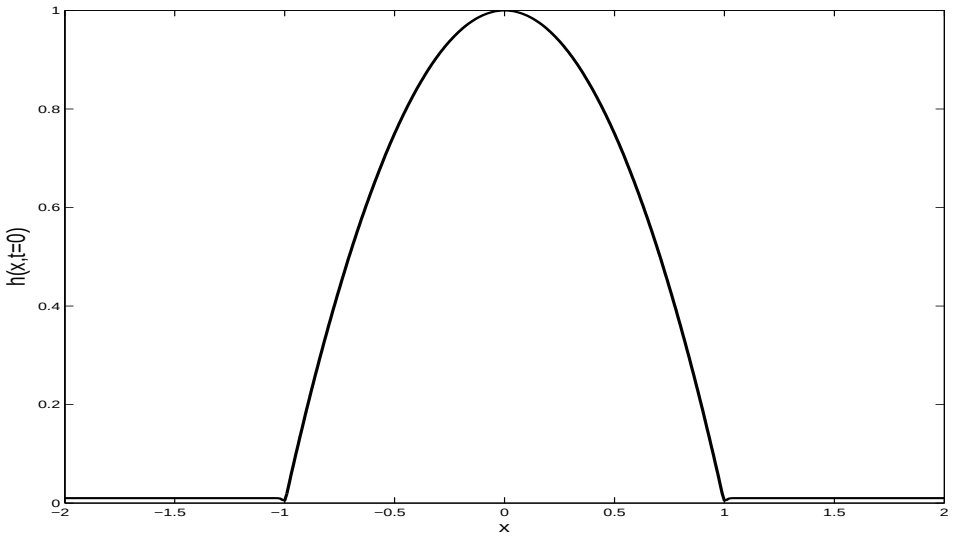

Figure 3.10: The initial shape of the parabolic drop connecting onto a precursor film of thickness $b = 0.01$ at the location $x = \pm 1$.

mesh followed by an adaptive moving mesh. The parameter values are: $Ca = 10^{-3}$, $b = 10^{-2}$ and $\theta = 90^o$ (so, $D = 0$). The upstream and downstream domain length is fixed at $L_1 = -2$ and $L_2 = 10$, respectively.

3.2.1 Numerical solution of Eqs. (3.1, 3.41) on a uniform mesh

We use the uniform mesh scheme Eq. (3.12) for discretising Eq. (3.1). As in the previous section, fictitious points are used, where necessary, near the ends of the domain and are derived from the BCs given by Eq. (3.41). The numerical solution procedure followed is similar to the previous section and is described in detail in §3.1.2.

We make the additional check to ensure that the drop volume is always conserved.

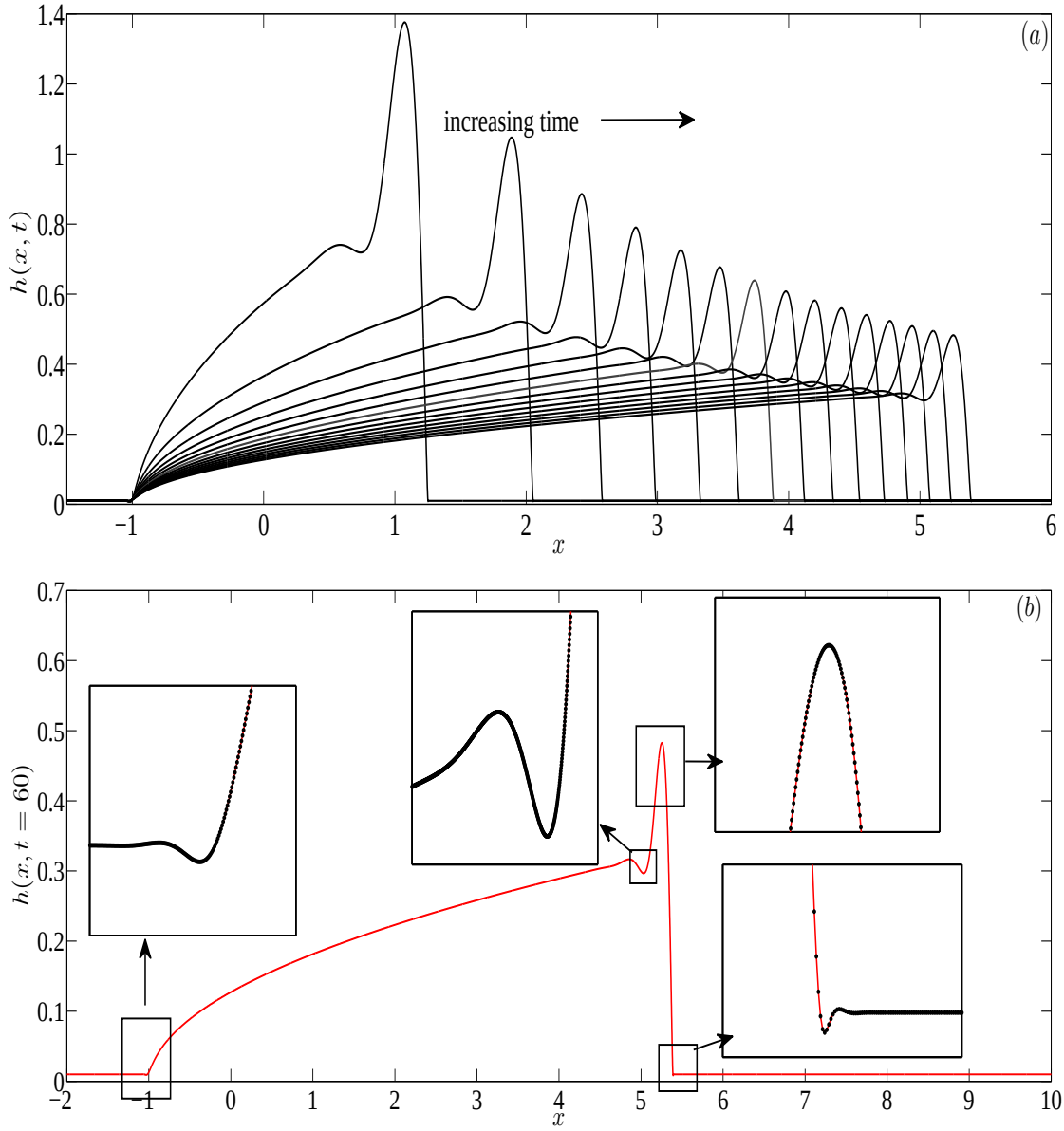

Figure 3.11: (a) Time evolution (time $t = 30 - 60$) of Eq (3.1) subject to the boundary conditions Eq. (3.41) using a uniform mesh scheme, and (b) a particular numerical solution at $t = 60$. The insets in (b) show zoomed-in capillary ridge and capillary waves at the leading and trailing edges. The parameter values are: $Ca = 10^{-3}$, $b = 10^{-2}$, $\theta = 90^o$ (so, $D = 0$) and $N = 12000$ (so, $\Delta x = 10^{-3}$). The upstream and downstream domain length is fixed at $L_1 = -2$ and $L_2 = 10$, respectively.

Figure 3.11(a) shows the time evolution of Eq (3.1) subject to the boundary conditions Eq. (3.41) and the initial condition Eq. (3.10) using the uniform mesh scheme. In the simulation shown the number of points used is $N = 12000$ (corresponding to $\Delta x = 10^{-3}$). The time t ranges between 30 to 60. We observe the development of the capillary ridge and capillary wave ahead and behind it near the drop's leading edge where it connects onto the precursor film (the wave ahead of the ridge has a much smaller width and is not seen on the scale shown). This is similar to the previous constant flux case but here the drop height steadily decreases in time as it spreads down the inclined substrate. We also observe a capillary wave that develops near the trailing edge of the drop. This is of similar width to the one near the leading edge and is also not seen on the scale shown. Figure 3.11(b) shows a particular numerical solution at time $t = 60$. The insets show the zoomed-in capillary ridge and capillary waves at the leading and trailing edges. We clearly observe that a large number of mesh points is required in order to fully resolve these regions which makes the numerical solution using a uniform mesh very computationally intensive. Figure 3.12(a, b) shows the same evolution as in figure 3.11 except that the number of mesh points $N = 600$ (corresponding to $\Delta x = 0.02$). We clearly observe that the capillary ridge and waves are under-resolved in comparison to the previous numerical solution. Next, we seek the numerical solution on an adaptive moving mesh.

For the adaptive moving mesh scheme we use Eq. (3.19), the curvature-based monitor function (see Eq. (3.39) with $\alpha = 100$) and MMPDE4 (see Eq. (3.30) with $\tau = 10^{-3}$).

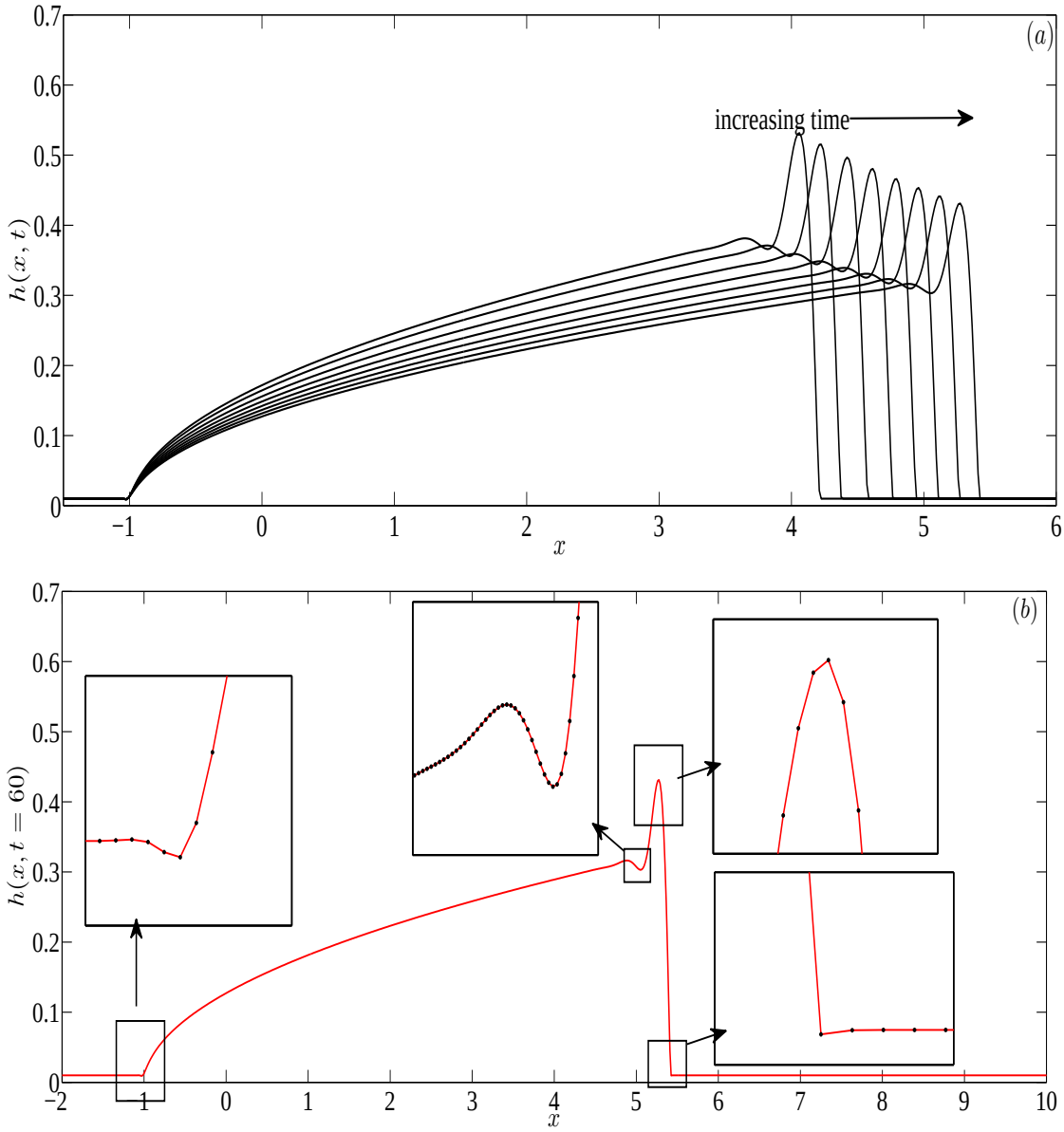

Figure 3.12: (a) Time evolution (time $t = 30 - 60$) of Eq (3.1) subject to the boundary conditions Eq. (3.41) using a uniform mesh scheme, and (b) a particular numerical solution at $t = 60$. The insets in (b) show zoomed-in capillary ridge and capillary waves at the leading and trailing edges. The parameter values are: $Ca = 10^{-3}$, $b = 10^{-2}$, $\theta = 90^o$ (so, $D = 0$) and $N = 600$ (so, $\Delta x = 0.02$). The upstream and downstream domain length is fixed at $L_1 = -2$ and $L_2 = 10$, respectively.

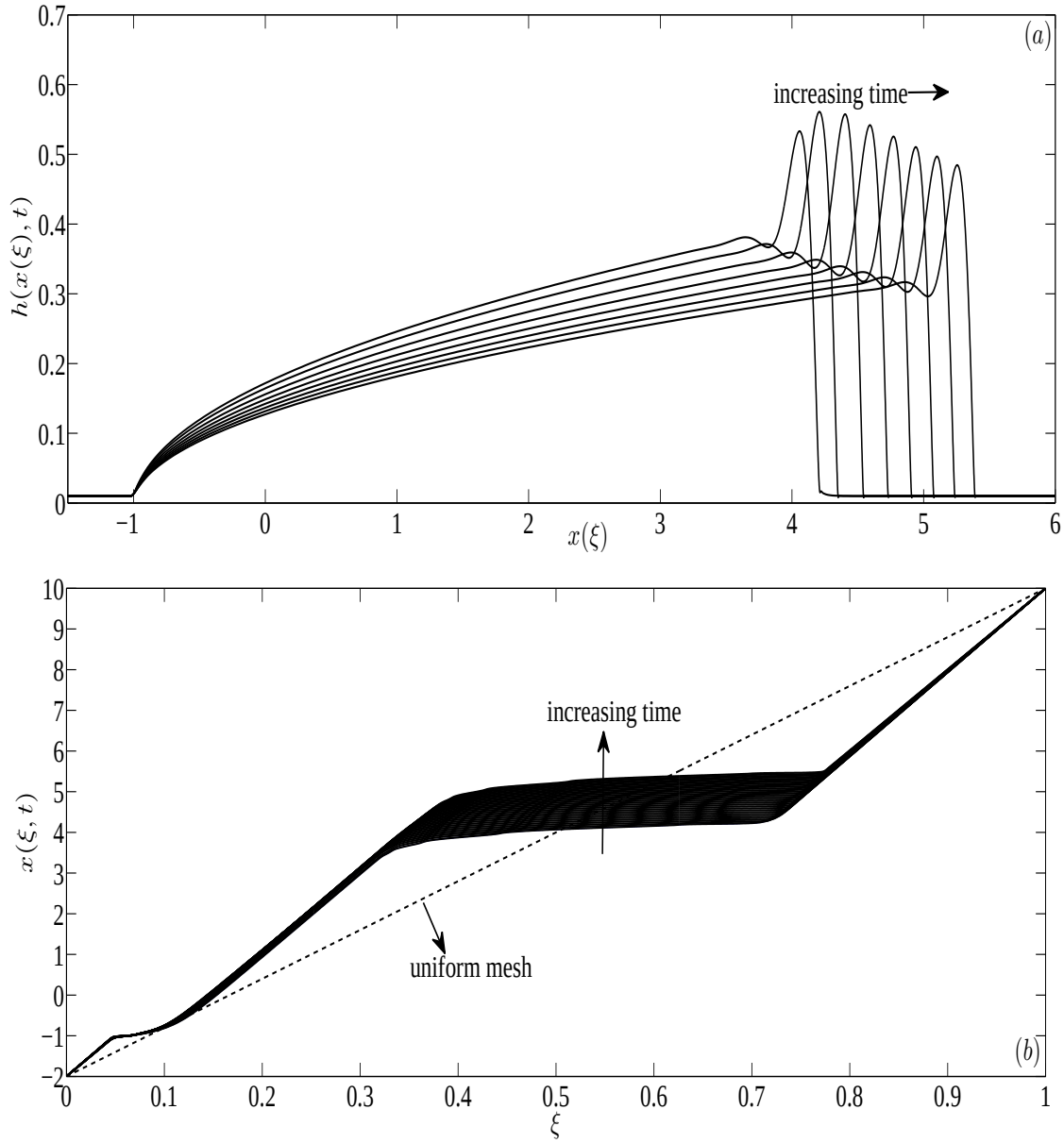

Figure 3.13: (a) Time evolution (time $t = 30 - 60$) of Eq (3.1) subject to the boundary conditions Eq. (3.41) using a moving adaptive mesh scheme, and (b) the equidistributing coordinate transformation $x(\xi)$ versus ξ. The dashed line shows the initial uniform mesh. The parameter values are: $Ca = 10^{-3}$, $b = 10^{-2}$, $\theta = 90^o$ (so, $D = 0$), $N = 600$ (so, initial $\Delta x = 0.02$), $\alpha = 100$ and $\tau = 10^{-3}$. The upstream and downstream domain length is fixed at $L_1 = -2$ and $L_2 = 10$, respectively.

A uniform mesh is used as the initial condition for the MMPDE. The numerical solution procedure followed is similar to the previous section and is described in detail in §3.1.3.

Figure 3.13(a) shows the time evolution of Eq (3.1) subject to the boundary conditions Eq. (3.41) and the initial condition Eq. (3.10) using the adaptive moving mesh scheme. In the simulation shown the number of points used is $N = 600$ (corresponding to initial $\Delta x = 0.02$). The time t ranges between 30 to 60. The solution structure is similar to that described for the uniform mesh simulation. Figure 3.13(b) shows the corresponding equidistributing coordinate transformation $x(\xi)$. The dashed line shows the initial uniform mesh. We observe from figure 3.13(b) that majority of mesh points are redistributed to the capillary ridge and waves near the leading edge of the drop where changes in curvature are very rapid. There is also redistribution of points near the trailing edge where there are also changes in curvature, although not as rapid as near the leading edge. The redistribution of points is observed to be more biased towards the leading edge of the front where there are rapid changes in curvature. To better control this redistribution so that the solution near trailing edge is also accurately resolved, we modify the parameters in the curvature-based monitor function as follows:

$$\rho(x,t) = \sqrt{1 + \alpha(x) h_{xx}^2}, \tag{3.43}$$

where

$$\alpha(x) = \begin{cases} 3, & \text{if } x \leq 1, \\ 10^{-3}, & \text{if } x > 1, \end{cases} \tag{3.44}$$

The higher value of α in the interval $[L_1, 1]$ ensures that more points are redistributed to the trailing edge region (compared to the previous simulation) while the lower value of α in the interval $[1, L_2]$ reduces the redistribution to the leading edge eventhough there are large changes in curvature there. We note that the choice of $x = 1$ is arbitrary in Eq. (3.44); any location at the upstream end of the trailing edge capillary wave would suffice. The piecewise constant values for α are based on some *a priori* knowledge of the values of the solution curvature at the leading and trailing edges of the drop.

Figure 3.14(a) shows the time evolution of Eq (3.1) subject to the boundary conditions Eq. (3.41) and the initial condition Eq. (3.10) using the adaptive moving mesh scheme using the curvature-based monitor function given by Eq. (**??**). In the simulation shown the number of points used is $N = 600$ (corresponding to initial $\Delta x = 0.02$). The time t ranges between 30 to 60. The solution structure, at least visually, looks more accurate than that in figure 3.13(a). Figure 3.14(b) shows the corresponding equidistributing coordinate transformation $x(\xi)$. The dashed line shows the initial uniform mesh. We clearly observe a marked change in the redistribution of points towards the trailing and leading edges in comparison to that shown in figure 3.13(b). The effect of this on the accuracy of the numerical solution is shown in figure 3.15(a, b), which plots the numerical solution at $t = 60$ for the adaptive moving mesh using the curvature-based monitor function given by Eq. (3.43) and Eq. (3.39) (with $\alpha = 100$), respectively. Inspection of the figures (see zoomed-in insets at leading and trailing edge of the drop), clearly show the well-resolved capillary wave near the trailing edge in (a) compared to

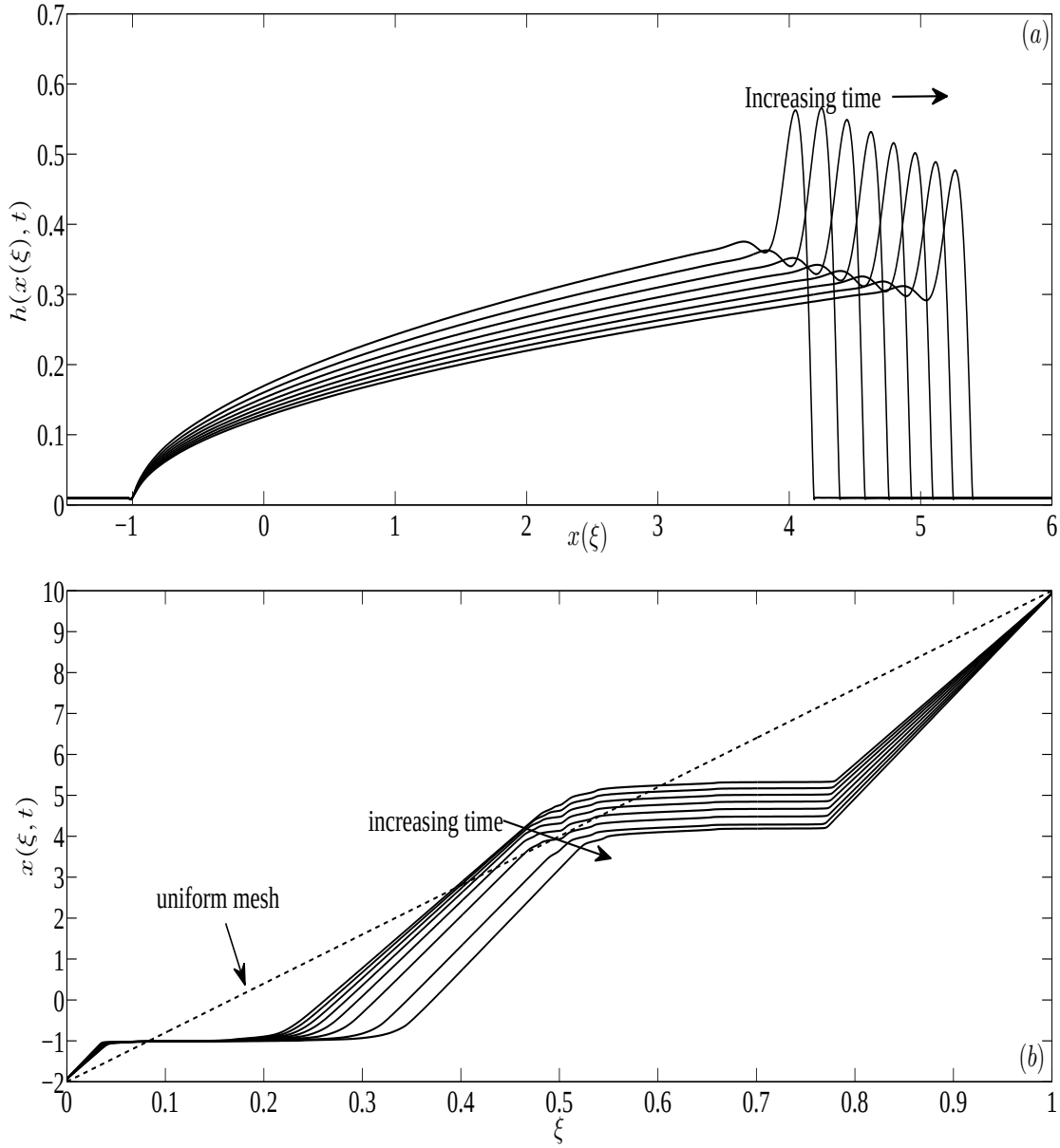

Figure 3.14: (a) Time evolution (time $t = 30 - 60$) of Eq (3.1) subject to the boundary conditions Eq. (3.41) using a moving adaptive mesh scheme, and (b) the equidistributing coordinate transformation $x(\xi)$ ploted against ξ, using the curvature-based monitor function given by Eq. (3.43). The dashed line in (b) shows the initial uniform mesh. The parameter values are: $Ca = 10^{-3}$, $b = 10^{-2}$, $\theta = 90^o$ (so, $D = 0$), $N = 600$ (so, initial $\Delta x = 0.02$), $\tau = 10^{-3}$ and α as given in Eq. (3.44) . The upstream and downstream domain length is fixed at $L_1 = -2$ and $L_2 = 10$, respectively.

that in (b); the leading edge structures are similar in both.

3.3 Conclusions

We have successfully implemented a moving adaptive mesh scheme based on moving mesh PDEs (MMPDEs) for a fourth order degenerate parabolic PDE. The main highlights of the results are shown in figure 3.9 and Table 3.2 which enables direct comparison with the uniform mesh scheme. We observe that the error for a fixed number of mesh points is always much lower for the adaptive moving mesh schemes compared to the corresponding uniform mesh scheme. Alternatively, for a prescribed error, the adaptive moving mesh scheme achieves this with far less number of points compared to the uniform mesh scheme. However, the adaptive moving mesh scheme takes much longer CPU times than the corresponding uniform mesh scheme. This is due to the extra mesh equation that needs to be solved along with the underlying PDE. This difference between CPU times is not that large if the desired error is not too small. We solve the underlying PDE and the mesh equation simultaneously which results in a larger bandwidth for the Jacobian matrix used in solving the linearised equations by the ODE solver. This could result in the increased CPU times observed. If this is the case then one could solve the PDE and mesh equation alternately, i.e., update the mesh solution at next time level based on the solution of the PDE at the current time level and then use this mesh solution to obtain the PDE solution at the next time level. Thus one solves a single PDE at each time step thereby reducing the bandwidth

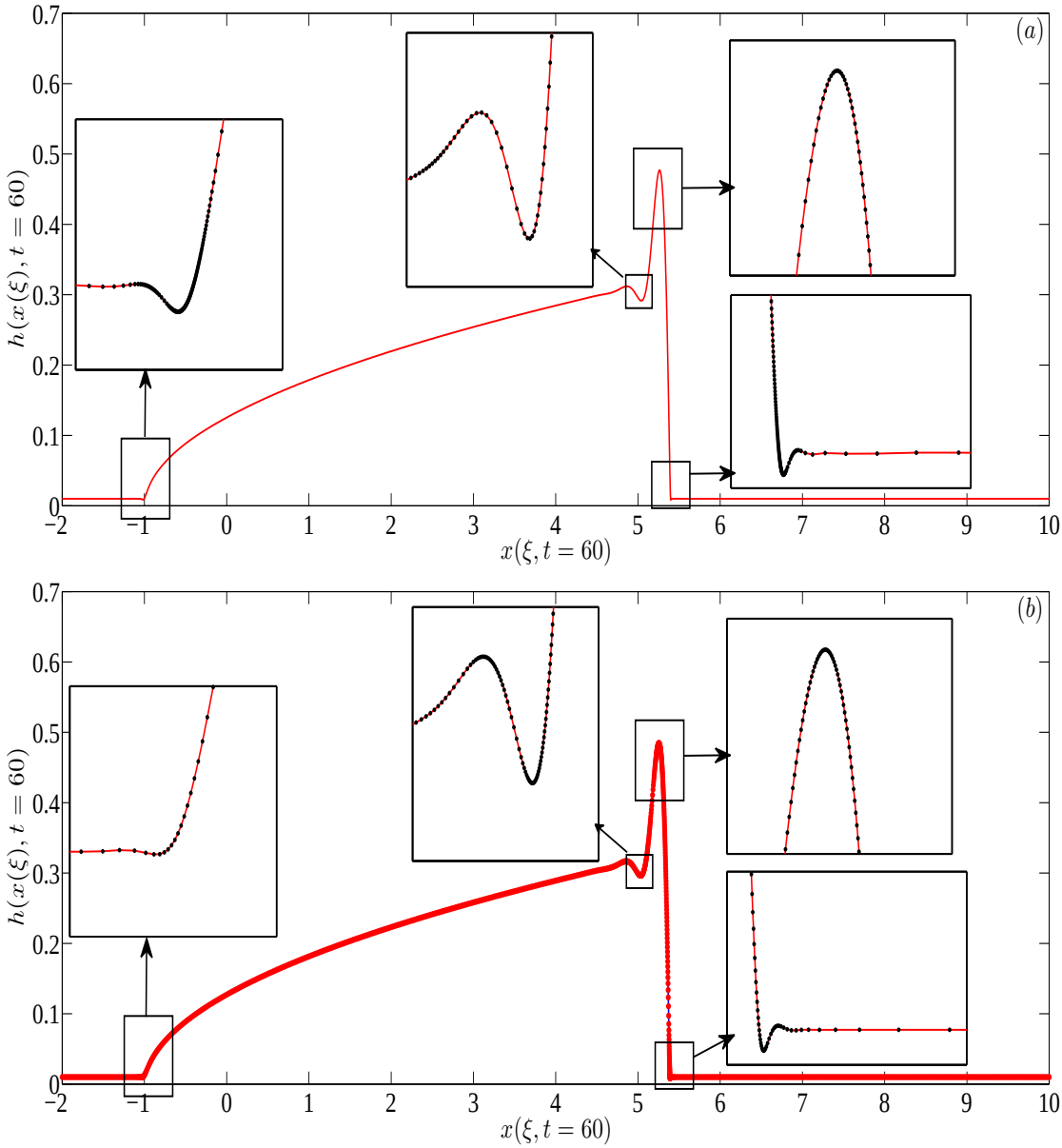

Figure 3.15: The numerical solution at $t = 60$ using the monitor function given by (a) Eq. (3.43), and (b) Eq. (3.39) with $\alpha = 100$. The parameter values are: $Ca = 10^{-3}$, $b = 10^{-2}$, $\theta = 90^o$ (so, $D = 0$), $N = 600$ (so, initial $\Delta x = 0.02$), $\alpha = 1$ and $\tau = 10^{-3}$. The upstream and downstream domain length is fixed at $L_1 = -2$ and $L_2 = 10$, respectively.

which could reduce the CPU times. This will be explored in future.

The other significant result is related to adaptation of the monitor function to accurately resolve the solution at multiple locations. In the constant flux BC case, we had to resolve the solution only around the capillary ridge region. In contrast for the constant volume BC case, we had to also resolve the capillary wave near the drop's trailing edge in addition to the capillary ridge region. We adapted the monitor function by choosing a piecewise constant (simplest option) weight parameter which allowed mesh points to be redistributed accordingly so as to resolve both regions. We note that this was not too difficult to implement for the constant volume BC since the trailing edge was almost stationary. However, it could be quite difficult if this was not the case. In any case, one would need to devise a similar strategy if the mesh requires to be adapted at multiple locations. One could also use a mixed monitor function, i.e., combination of curvature-based and arc-length monitor functions. These would need to be explored in future. This also highlights one of the drawbacks of r-adaptive methods which redistribute a fixed number of points. One would need to start with a large number of points if there are multiple locations to resolve along with a judiciously chosen monitor function. In this case, it might be more advantageous to consider an hp-refinement method.

In conclusion, our results indicate great promise for MMPDEs in thin-film flow problem. Although we have only considered a specific form of the underlying PDE, the general framework presented in this chapter can be utilised for any thin-film equation of the

form considered here.

Chapter 4

Numerical Solution of 1D Coupled PDEs on an Adaptive Moving Mesh

4.1 PDEs with constant flux boundary conditions

Consider the one-dimensional coupled system of two fourth and second order nonlinear PDEs described in Chapter 1,

$$h_t + \left[Ca\ \frac{h^3}{3}\ h_{xxx}\ - G\cos(\theta)\frac{h^3}{3}h_x\ - \frac{h^2}{2}\Gamma_x + G\sin(\theta)\ \frac{h^3}{3}\right]_x\ =\ 0, \tag{4.1}$$

$$\Gamma_t + \left[Ca\ \frac{h^2}{2}\Gamma\ h_{xxx}\ - G\cos(\theta)\frac{h^2}{2}\Gamma\ h_x\ - h\Gamma\Gamma_x + G\sin(\theta)\ \frac{h^2}{2}\Gamma\right]_x\ = \delta\Gamma_{xx}, \tag{4.2}$$

$(x,t) \in [0,L] \times [0,T_f]$.

Here, $h = h(x,t)$ and $\Gamma = \Gamma(x,t)$ are the film height and surfactant concentration, respectively, x is the spatial variable and t is time. The dimensionless parameter $Ca \ll 1$, is the Capillary number, G is a Bond number, δ, is the inverse of the Péclet number, L is a domain length and T_f is a specific time.

Equations (4.1,4.2) are supplemented by six boundary conditions (BCs) in x. These are:

$$h(0,t) = 1,\ h_{xxx}(0,t) = 0,\ \Gamma(0,t) = 1,\ h(L,t) = b,\ h_{xxx}(L,t) = 0,\ \Gamma(L,t) = 0, \quad (4.3)$$

where $b \ll 1$, is the precursor film thickness. These BCs represent a fixed fluid and surfactant flux source far upstream and a flat precursor film and zero surfactant concentration far downstream. We choose the following initial condition for h and Γ following [35, 37, 39]:

$$h(x,0) = (1-x^2)H(1-x) + bH(x-1),\ \ \Gamma(x,0) = H(1-x) - H(-x-1), \quad (4.4)$$

where $H(x)$ is the Heaviside function. Figure 4.1 shows the initial condition using Eq. (4.4) for $b = 10^{-2}$. The initial condition for h represents a drop connecting onto a thin precursor film and for the surfactant concentration represents a large gradient at the location where the drop connects onto the precursor film.

We now seek the numerical solution of Eqs. (4.1,4.2) with the boundary and the initial conditions using finite difference methods. The numerical method is performed

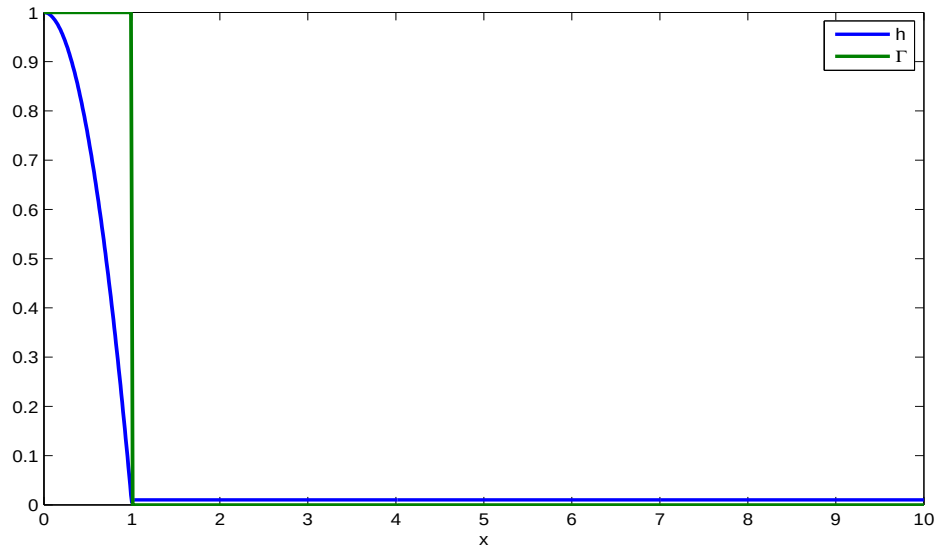

Figure 4.1: The initial condition for the film thickness h using Eq. (4.4). The precursor thickness $b = 10^{-2}$.

first on a uniform mesh and then on an adaptive moving mesh. Levy and co-workers [60, 61] have analysed the travelling wave structure of this problem and have shown the existence of multiple waves travelling with the same speed. In principle, one can use this multiple travelling wave structure to compare the convergence and accuracy of the numerical solution. We do not do this here but as an alternative use the numerical solution using a uniform mesh with sufficiently small Δx as an exact solution to compare with that using an adaptive mesh.

4.1.1 Numerical solution of Eqs. (4.1, 4.2, 4.3) on a uniform mesh

The domain $[0, L]$ is uniformly divided into $N+1$ discrete points as follows: $x_j = (j-1)\,\Delta x,\ \forall\ x_j \in [0, L],\quad j = 1, 2, ..., N+1$, where $\Delta x = \frac{L}{N}$ is the width of each sub-interval. We discretise the spatial derivatives appearing in Eq. (4.1,4.2) using finite differences; the time derivative is kept continuous. For convenience, we let $\alpha_1 = G\sin(\theta)$ and $\beta_1 = G\cos(\theta)$ in what follows.

The spatial discretisation is done as follows. We define a forward and a backward difference as:

$$(h,\Gamma)_{x,j+\frac{1}{2}} = \frac{(h,\Gamma)_{j+1} - (h,\Gamma)_j}{\Delta x} + O(\Delta x), \quad (h,\Gamma)_{\bar{x},j+\frac{1}{2}} = \frac{(h,\Gamma)_j - (h,\Gamma)_{j-1}}{\Delta x} + O(\Delta x),$$

respectively.

Thus, a semi-discretisation of Eq. (4.1,4.2), keeping the time derivative continuous, is

$$\begin{aligned} &h_{t,j} + \frac{Ca}{3}\left[\frac{(h^3)_{j+\frac{1}{2}}\,(h_{x\bar{x}x})_{j+\frac{1}{2}} - (h^3)_{j-\frac{1}{2}}\,(h_{x\bar{x}x})_{j-\frac{1}{2}}}{\Delta x}\right] \\ &- \frac{\beta_1}{3}\left[\frac{(h^3)_{j+\frac{1}{2}}\,(h_x)_{j+\frac{1}{2}} - (h^3)_{j-\frac{1}{2}}\,(h_x)_{j-\frac{1}{2}}}{\Delta x}\right] - \frac{1}{2}\left[\frac{(h^2)_{j+\frac{1}{2}}(\Gamma_x)_{j+\frac{1}{2}} - (h^2)_{j-\frac{1}{2}}(\Gamma_x)_{j-\frac{1}{2}}}{\Delta x}\right] \\ &+ \frac{\alpha_1}{3}\left[\frac{(h^3)_{j+\frac{1}{2}} - (h^3)_{j-\frac{1}{2}}}{3\Delta x}\right] = 0, j = 2, ..., N. \end{aligned} \tag{4.5}$$

$$\Gamma_{t,j} + \frac{Ca}{2}\left[\frac{(h^2\Gamma)_{j+\frac{1}{2}}(h_{x\bar{x}x})_{j+\frac{1}{2}} - (h^2\Gamma)_{j-\frac{1}{2}}(h_{x\bar{x}x})_{j-\frac{1}{2}}}{\Delta x}\right]$$
$$- \frac{\beta_1}{2}\left[\frac{(h^2\Gamma)_{j+\frac{1}{2}}(h_x)_{j+\frac{1}{2}} - (h^2\Gamma)_{j-\frac{1}{2}}(h_x)_{j-\frac{1}{2}}}{\Delta x}\right] - \left[\frac{(h\Gamma)_{j+\frac{1}{2}}(\Gamma_x)_{j+\frac{1}{2}} - (h\Gamma)_{j-\frac{1}{2}}(\Gamma_x)_{j-\frac{1}{2}}}{\Delta x}\right]$$
$$+ \frac{\alpha_1}{2}\left[\frac{(h^2\ \Gamma)_{j+\frac{1}{2}} - (h^2\ \Gamma)_{j-\frac{1}{2}}}{3\Delta x}\right] = \delta\frac{(\Gamma_x)_{j+\frac{1}{2}} - (\Gamma_x)_{j-\frac{1}{2}}}{\Delta x}, \quad j = 2, ..., N. \tag{4.6}$$

In the above, we approximate terms denoted by $(a)_{j+\frac{1}{2}}$ using spatial averages: $(a)_{j+\frac{1}{2}} = \frac{a_{j+1}+a_j}{2}$ or $(a)_{j+\frac{1}{2}} = \left(\frac{a_{j+1}+a_j}{2}\right)$. A similar approximation is used for terms denoted by $(a)_{j-\frac{1}{2}}$.

Evaluating Eq. (4.5,4.6) at $j = 2, N$ require fictitious points which are obtained by discretising the boundary conditions $h_{xxx} = 0$ at $x = 0, L$ using centred finite differences.

The boundary conditions $h(0,t) = \Gamma(0,t) = 1$, $h(L,t) = b$ and $\Gamma(L,t) = 0$ are replaced by their ODE form:

$$h_{t,1} = h_{t,N+1} = \Gamma_{t,1} = \Gamma_{t,N+1} = 0. \tag{4.7}$$

This semi-discretisation scheme in Eqs. (4.5,4.6) is second order accurate in x. The scheme for h in Eq. (4.5) uses a 5-point stencil (has a smaller bandwidth in comparison with a 7-point stencil if using a centred finite difference scheme). The scheme for Γ in Eq. (4.5) uses a standard 3-point stencil for the second derivative in Γ. Equations (4.5,4.6,4.7) form a system of $2(N+1)$ ordinary differential equations for the solution $h_1, \ldots, h_{N+1}$ and $\Gamma_1, \ldots, \Gamma_{N+1}$ with initial condition given by Eq. (4.1). These are solved using the stiff ODE solver DASSL [72] which uses backward dif-

ferentiation formulae (BDFs) to approximate the time derivative and the resulting nonlinear system of equations are solved using Newton's method. We use a staggered system for numbering the unknowns, i.e., $h_1, \Gamma_1, h_2, \Gamma_2, \ldots, h_{N+1}, \Gamma_{N+1}$ which provides a smaller bandwidth for the Jacobian matrix (note: the bandwidth=9 for the semi-discretisation in Eq. (4.5,4.6)). This is in comparison to, for example, the numbering $h_1, h_2, \ldots, h_{N+1}, \Gamma_1, \Gamma_2, \ldots, \Gamma_{N+1}$, which although sparse has a much bigger bandwidth. This significantly improves the performance of the solver. We note here that the stiff solver DASPK [12] was also used which utilises an iterative scheme based on Krylov subspace methods for the solution of the linearised system (DASSL uses a direct solver) including preconditioning using Incomplete LU decomposition of the Jacobian matrix. This had no significant influence on the performance in comparison to DASSL, at least for the 1D problem considered in this chapter.

Figures 4.2(a, b) illustrate the time evolution of h and Γ, respectively, to a travelling wave solution using the above uniform mesh scheme. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $L = 40$ and $\Delta x = 10^{-3}$ (so, $N = 4 \times 10^4$). The results are shown for time t ranging between 0 to 90. The solution for h shows a multiple travelling wave structure characterised by a capillary ridge and capillary waves near it's leading edge. In addition, a "step"-like structure where $h \approx 2b$ develops ahead of this leading edge [35, 37, 39, 60, 61]. The solution for Γ, although continuous, displays rapid variation in it's gradient, particularly, near it's leading edge.

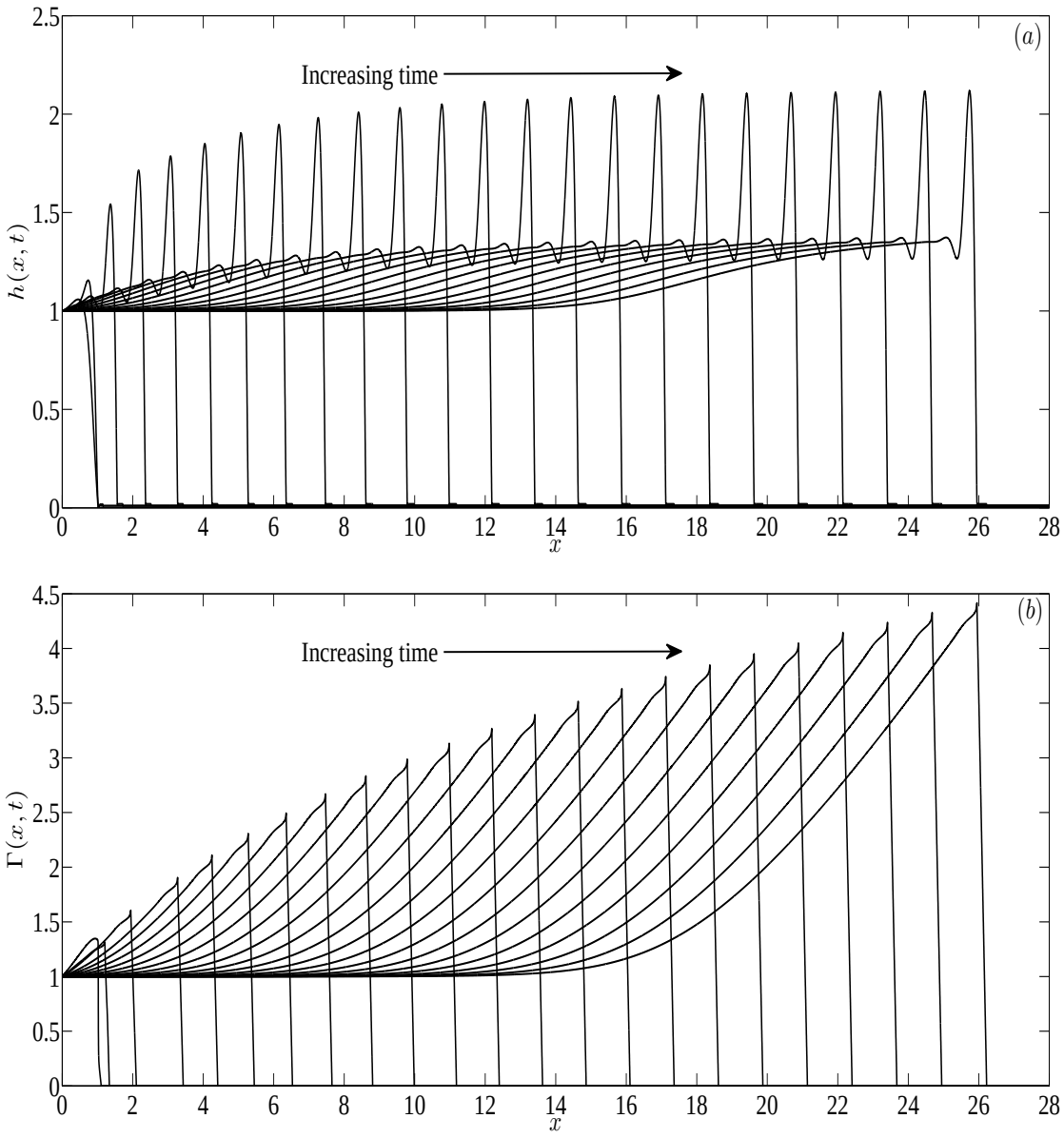

Figure 4.2: Time evolution of (a) $h(x,t)$ and (b) $\Gamma(x,t)$, using the uniform mesh scheme given in Eqs. (4.5,4.6). The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $L = 40$ and $\Delta x = 10^{-3}$ (so, $N = 4 \times 10^4$). The results are shown for time t ranging between 0 to 90.

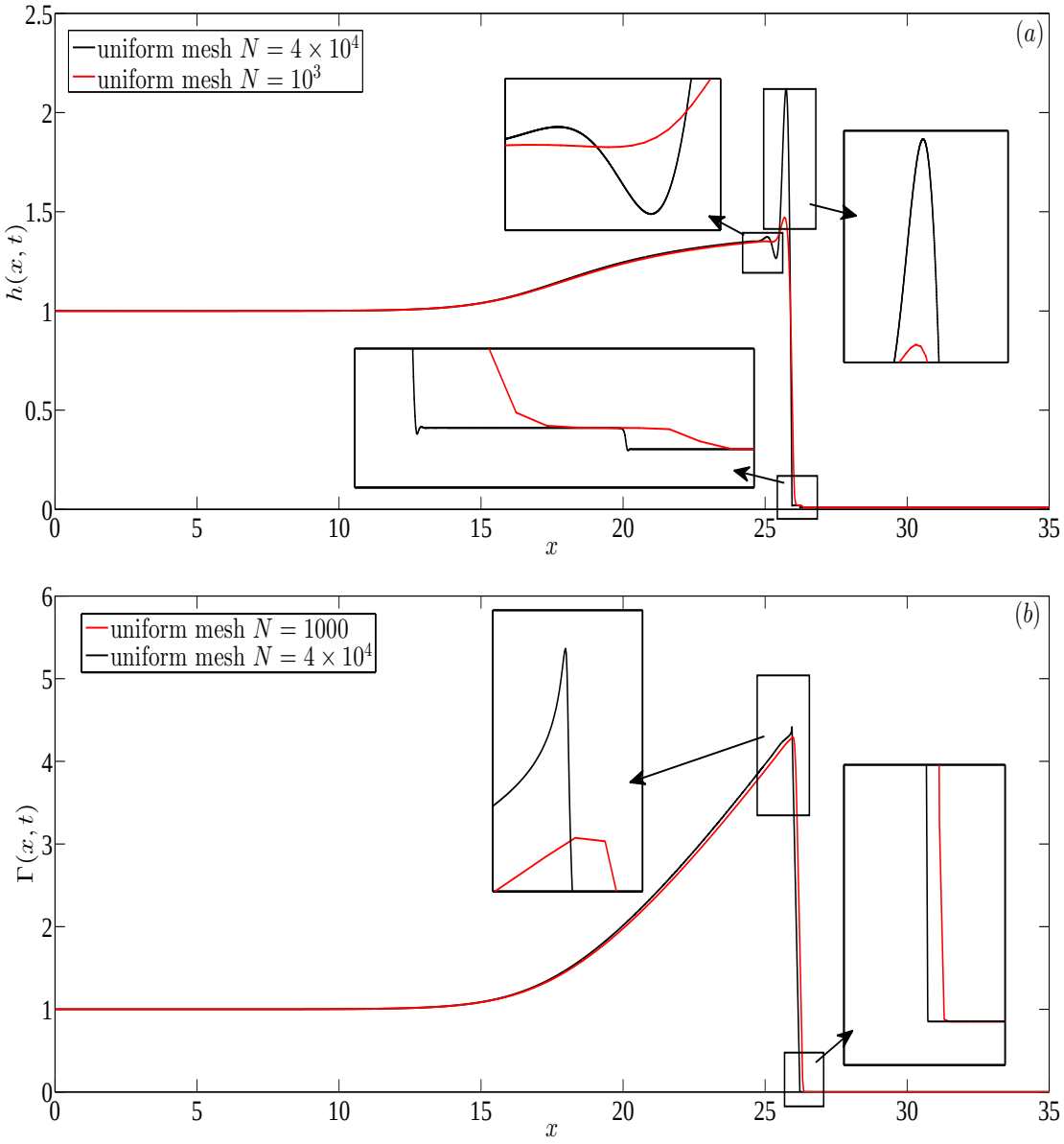

Figure 4.3: The numerical solution for (a) h and (b) Γ at $t = 90$, using the uniform mesh scheme given in Eqs. (4.5,4.6) with $N = 1000$ points or $\Delta x = 0.04$ (solid red line) and $N = 4 \times 10^4$ or $\Delta x = 10^{-3}$ points (solid black line). The insets show zoomed-in solution characteristics near the leading edge of h and Γ. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$ and $L = 40$.

Figures 4.3(a, b) shows a particular numerical solution at time $t = 90$ for h and Γ, repsectively. The insets show the characteristics of the wave-like structures described above in more detail, including the step-like structure. We also compare two numerical solutions corresponding to $N = 4\times10^4$ or $\Delta x = 10^{-3}$ (solid black line) and $N = 1000$ or $\Delta x = 0.04$ (solid red line). Inspection of these figures reveal that the numerical solution has several regions (see insets) that are not appropriately resolved with $N = 1000$ points ($\Delta x = 0.04$). These regions require a much finer mesh (as seen with $N = 4\times10^4$ points or $\Delta x = 10^{-3}$) to be adequately resolved using a uniform mesh. We have checked the accuracy and convergence of the numerical solution as $\Delta x \to 0$ (not shown here) and confirm that the solution with $N = 4\times10^4$ points or $\Delta x = 10^{-3}$ is sufficiently accurate. This makes the numerical solution on a uniform mesh computationally intensive and hence the use of an adaptive moving mesh would be beneficial in this respect.

4.1.2 Numerical solution of Eqs. (4.1, 4.2, 4.3) on an adaptive moving mesh

We now seek the numerical solution of Eqs. (4.1,4.2,4.4) using an adaptive moving mesh method. The adaptive moving mesh method uses a coordinate transformation from the computational domain with coordinate ξ, to the physical domain with the coordinate x:

$$x = x(\xi)\colon \Omega_c \equiv [0, 1] \to \Omega_p \equiv [0, L].$$

Ω_c is called the computational domain and Ω_p is the physical domain. Then the solution can be written as

$$h(x,t) = h(x(\xi,t),t), \quad \Gamma(x,t) = \Gamma(x(\xi,t),t).$$

A moving mesh associated with the solutions h and Γ is described as

$$\mathcal{J}_{h,\Gamma}(t)\colon x_j(\xi) = x(\xi_j,t),\ j = 1,\cdots,N+1 \tag{4.8}$$

where the boundary nodes are given by

$$x_1 = 0, \quad x_{N+1} = L. \tag{4.9}$$

A uniform mesh on the computational domain is described as

$$\mathcal{J}^c_{h,\Gamma}(t)\colon \xi_j = \frac{(j-1)}{N},\ j = 1,\cdots,N+1. \tag{4.10}$$

Using the chain rule we have the following transformations

$$h_x = \frac{h_\xi}{x_\xi}, \quad h_t = h_t - \frac{h_\xi}{x_\xi}\,x_t, \quad \Gamma_x = \frac{\Gamma_\xi}{x_\xi}, \quad \Gamma_t = \Gamma_t - \frac{\Gamma_\xi}{x_\xi}\,x_t. \tag{4.11}$$

Using the above, Eqs. (4.1,4.2) can be written as

$$h_t - \frac{h_\xi}{x_\xi}x_t = -\frac{Q_\xi}{x_\xi}, \quad \Gamma_t - \frac{\Gamma_\xi}{x_\xi}x_t = -\frac{S_\xi}{x_\xi}, \tag{4.12}$$

$$Q = Ca\left(\frac{h^3}{3}\right)\frac{1}{x_\xi}\left(\frac{\left(\frac{h_\xi}{x_\xi}\right)_\xi}{x_\xi}\right)_\xi - \beta_1\left(\frac{h^3}{3}\right)\frac{h_\xi}{x_\xi} - \left(\frac{h^2}{2}\right)\frac{\Gamma_\xi}{x_\xi} + \alpha_1\frac{h^3}{3}, \tag{4.13}$$

$$S = Ca\left(\frac{h^2\Gamma}{2}\right)\frac{1}{x_\xi}\left(\frac{\left(\frac{h_\xi}{x_\xi}\right)_\xi}{x_\xi}\right)_\xi - \beta_1\left(\frac{h^2\Gamma}{2}\right)\frac{h_\xi}{x_\xi} - (h\Gamma+\delta)\frac{\Gamma_\xi}{x_\xi} + \alpha_1\frac{h^2\Gamma}{2}. \tag{4.14}$$

A *conservative* semi-discretisation scheme for the spatial derivatives in Eq. (4.12) on the uniform mesh $\mathcal{J}_h^c(t)$ using centred finite differences can be written as, keeping the time derivative continuous,

$$h_{t,j} - \frac{(h_{j+1}-h_{j-1})}{(x_{j+1}-x_{j-1})}x_{t,j} = -2\frac{Q_j - Q_{j-1}}{x_{j+1}-x_{j-1}}, \tag{4.15}$$

$$\Gamma_{t,j} - \frac{(\Gamma_{j+1}-\Gamma_{j-1})}{(x_{j+1}-x_{j-1})}x_{t,j} = -2\frac{S_j - S_{j-1}}{x_{j+1}-x_{j-1}}, \text{ where} \tag{4.16}$$

$$Q_j = Ca\left(\frac{h^3}{3}\right)_{j+1/2}\frac{1}{x_{\xi,j}}\left(\frac{\left(\frac{h_\xi}{x_\xi}\right)_\xi}{x_\xi}\right)_{\xi,j} - \beta_1\left(\frac{h^3}{3}\right)_{j+1/2}\left(\frac{h_\xi}{x_\xi}\right)_{,j} + \alpha_1\left(\frac{h^3}{3}\right)_{j+1/2}, \tag{4.17}$$

$$S_j = Ca\left(\frac{h^2\Gamma}{2}\right)_{j+1/2}\frac{1}{x_{\xi,j}}\left(\frac{\left(\frac{h_\xi}{x_\xi}\right)_\xi}{x_\xi}\right)_{\xi,j} - \beta_1\left(\frac{h^2\Gamma}{2}\right)_{j+1/2}\left(\frac{h_\xi}{x_\xi}\right)_{,j} - \left[(h\Gamma)_{j+1/2}+\delta\right]\left(\frac{\Gamma_\xi}{x_\xi}\right)_{,j} + \alpha_1\left(\frac{h^2\Gamma}{2}\right)_{j+1/2}, \tag{4.18}$$

$$j = 2,\ldots,N.$$

The spatial derivatives appearing in Q_j and S_j are discretised as follows:

$$\frac{1}{x_{\xi,j}}\left(\frac{\left(\frac{h_\xi}{x_\xi}\right)_\xi}{x_\xi}\right)_{\xi,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)\left(\frac{1}{x_{\xi,j+1}}\left(\frac{h_\xi}{x_\xi}\right)_{\xi,j+1} - \frac{1}{x_{\xi,j}}\left(\frac{h_\xi}{x_\xi}\right)_{\xi,j}\right), \quad \text{where}$$
$$\frac{1}{x_{\xi,j}}\left(\frac{h_\xi}{x_\xi}\right)_{\xi,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)\left(\frac{h_{j+1}-h_j}{x_{j+1}-x_j} - \frac{h_j-h_{j-1}}{x_j-x_{j-1}}\right),$$
$$\left(\frac{h_\xi}{x_\xi}\right)_{,j} = \frac{h_{j+1}-h_j}{x_{j+1}-x_j}, \quad \left(\frac{\Gamma_\xi}{x_\xi}\right)_{,j} = \frac{\Gamma_{j+1}-\Gamma_j}{x_{j+1}-x_j}, \quad j=1,\ldots,N. \tag{4.19}$$

Approximations to terms of the form $a_{j+1/2}$ and $a_{j-1/2}$ are the same as defined previously.

We can also write a *non-conservative* semi-discretisation scheme for the spatial derivatives in Eqs. (4.1,4.2). To do this, we write Eqs. (4.1,4.2) in the following form:

$$h_t + Ca\left[\frac{h^3}{3}p_x\right]_x - \beta_1\left[\frac{h^3}{3}h_x\right]_x - \left[\frac{h^2}{2}\Gamma_x\right]_x + \alpha_1\left[\frac{h^3}{3}\right]_x = 0, \tag{4.20}$$
$$\Gamma_t + Ca\left[\frac{h^2\Gamma}{2}p_x\right]_x - \beta_1\left[\frac{h^2\Gamma}{2}h_x\right]_x - [(h\Gamma+\delta)\Gamma_x]_x + \alpha_1\left[\frac{h^2\Gamma}{2}\right]_x = 0, \quad p = h_{xx}. \tag{4.21}$$

A *non-conservative* semi-discretisation scheme for a function of the form $[au_x]_x$ on the uniform mesh $\mathcal{J}_h^c(t)$ using finite differences can be written as:

$$[au_x]_{x,j} = \frac{1}{x_{\xi,j}}\left[\frac{au_\xi}{x_\xi}\right]_{\xi,j} = \frac{1}{x_{\xi,j}}\left[\frac{[au_\xi]_{\xi,j}}{x_{\xi,j}} - \frac{1}{x_{\xi,j}^2}a_j u_{\xi,j} x_{\xi\xi,j}\right] =$$
$$\left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2\left[a_{j+\frac{1}{2}}(u_{j+1}-u_j) - a_{j-\frac{1}{2}}(u_j-u_{j-1})\right]$$
$$- a_j\left(\frac{u_{j+1}-u_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right). \tag{4.22}$$

Using the above, a *non-conservative* semi-discretisation scheme for the spatial derivatives in Eqs. (4.20,4.21) on the uniform mesh $\mathcal{J}_h^c(t)$ using finite differences can be written as, keeping the time derivative continuous,

$$\begin{aligned}
&h_{t,j} + Ca\left[\frac{h^3}{3}p_x\right]_{x,j} - \beta_1\left[\frac{h^3}{3}h_x\right]_{x,j} - \left[\frac{h^2}{2}\Gamma_x\right]_{x,j} + \alpha_1\left[\frac{\hat{h}^3}{3}\right]_{x,j} = 0,\\
&\Gamma_{t,j} + Ca\left[\frac{h^2\Gamma}{2}p_x\right]_{x,j} - \beta_1\left[\frac{h^2\Gamma}{2}h_x\right]_{x,j} - \left[(h\Gamma+\delta)\Gamma_x\right]_{x,j} + \alpha_1\left[\frac{\hat{h}^3}{3}\right]_{x,j} = 0, \qquad (4.23)\\
&p_j = h_{xx,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2\left[h_{j+1}-2h_j+h_{j-1}\right]\\
&-\left(\frac{h_{j+1}-h_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right), \qquad (4.24)\\
&[ap_x]_{x,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2\left[a_{j+\frac{1}{2}}(p_{j+1}-p_j) - a_{j-\frac{1}{2}}(p_j-p_{j-1})\right]\\
&-a_j\left(\frac{p_{j+1}-p_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right), \qquad (4.25)\\
&[ah_x]_{x,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2\left[a_{j+\frac{1}{2}}(h_{j+1}-h_j) - a_{j-\frac{1}{2}}(h_j-h_{j-1})\right]\\
&-a_j\left(\frac{h_{j+1}-h_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right), \qquad (4.26)\\
&[a\Gamma_x]_{x,j} = \left(\frac{2}{x_{j+1}-x_{j-1}}\right)^2\left[a_{j+\frac{1}{2}}(\Gamma_{j+1}-\Gamma_j) - a_{j-\frac{1}{2}}(\Gamma_j-\Gamma_{j-1})\right]\\
&-a_j\left(\frac{\Gamma_{j+1}-\Gamma_{j-1}}{x_{j+1}-x_{j-1}}\right)\left(\frac{x_{j+1}-2x_j+x_{j-1}}{(x_{j+1}-x_j)(x_j-x_{j-1})}\right), \qquad (4.27)\\
&[h^3]_x = 2\left(\frac{h^3_{j+1/2}-h^3_{j-1/2}}{x_{j+1}-x_{j-1}}\right). \qquad (4.28)
\end{aligned}$$

Approximations to terms of the form $a_{j+1/2}$ and $a_{j-1/2}$ are the same as defined previously.

The boundary conditions $h(0,t) = \Gamma(0,t) = 1$, $h(L,t) = b$ and $\Gamma(L,t) = 0$ are replaced

by their ODE form:

$$h_{t,1} = 0, \;\; h_{t,N+1} = 0, \;\; \Gamma_{t,1} = 0, \;\; \Gamma_{t,N+1} = 0. \tag{4.29}$$

The equidistributing coordinate transformation $x = x(\xi, t)$ is obtained by solving the moving mesh PDEs (MMPDEs). Based on the results in Chapter 3 (§3.1.3) we choose MMPDE 4. Recall the semi-discretisation of MMPDE 4:

$$\begin{aligned} \text{MMPDE 4}: \quad & (\rho x_{t\xi})_\xi = -\frac{1}{\tau}(\rho x_\xi)_\xi, \\ \text{semi-discretisation}: \quad & \tau[(\rho_{j+1}+\rho_j)(x_{t,j+1}-x_{t,j}) - (\rho_{j-1}+\rho_j)(x_{t,j}-x_{t,j-1})] = \\ & -[(\rho_{j+1}+\rho_j)(x_{j+1}-x_j) - (\rho_{j-1}+\rho_j)(x_j - x_{j-1})], \\ & j = 2, \ldots, N. \end{aligned} \tag{4.30}$$

Here, $\rho(x,t)$ is a monitor function (described below), $\Delta\xi = \frac{1}{N}$ is the mesh size in the computational domain and $\tau > 0$ is a user-specified relaxation parameter.

The boundary conditions $x(0,t) = 0$ and $x(1,t) = L$ are replaced by their ODE form:

$$x_{t,1} = 0, \;\; x_{t,N+1} = 0. \tag{4.31}$$

The initial condition

$$x(\xi, 0) = L\xi, \tag{4.32}$$

which represents a uniform initial mesh on the physical domain $\Omega_p \equiv [0, L]$. We note here that a non-uniform initial mesh was also used which was obtained by solving in pseudo-time the chosen MMPDE (with the uniform mesh as the initial condition) with h and Γ fixed (hence, the monitor function ρ is also fixed) at its initial condition given by Eq. (4.4). This did not have any significant influence on the solution or the performance of the solver in comparison to the uniform initial mesh.

Finally, we choose the curvature-based monitor function based on the results in Chapter 3 (§3.1.3). We note here that the monitor functions described in chapter 3 are based on the solution having only a single component, i.e., only h. When the solution has multi-components (as in the case here where the solution has two components h and Γ), these monitor functions need to be adapted to account for rapid variations in the other components (which need not necessarily be aligned to variations in one component). Therefore, the curvature-based monitor function Eq. (3.39) is modified by the addition of the curvature in Γ and discretised using finite differences as:

$$\text{modified Curvature-based monitor function :} \quad \rho(x,t) = (1 + \beta|h_{xx}|^2 + \omega|\Gamma_{xx}|^2)^{\frac{1}{4}},$$

$$\text{semi-discretisation :} \quad \rho(x_j,t) = (1 + \beta|h_{xx,j}|^2 + \omega|\Gamma_{xx,j}|^2)^{\frac{1}{4}},$$

$$(h,\Gamma)_{xx,j} := \begin{cases} \frac{2}{(x_{j+1}-x_{j-1})}\left[\frac{((h,\Gamma)_{j+1}-(h,\Gamma)_j)}{(x_{j+1}-x_j)} - \frac{((h,\Gamma)_j-(h,\Gamma)_{j-1})}{(x_j-x_{j-1})}\right], & j = 2, ..., N, \\ \frac{2[(x_2-x_1)((h,\Gamma)_3-(h,\Gamma)_1)-(x_3-x_1)((h,\Gamma)_2-(h,\Gamma)_1)]}{(x_3-x_1)(x_3-x_2)(x_2-x_1)}, & j = 1, \\ \frac{2[(x_N-x_{N+1})((h,\Gamma)_{N-1}-(h,\Gamma)_{N+1})-(x_{N-1}-x_{N+1})((h,\Gamma)_N-(h,\Gamma)_{N+1})]}{(x_{N-1}-x_{N+1})(x_N-x_{N+1})(x_{N-1}-x_N)}, & j = N+1. \end{cases} \tag{4.33}$$

Here β and ω are user-specified parameters. We also use the same smoothing process for the discretised monitor function as described in Chapter 3 (§3.1.3) using Eq. (3.40).

Eqs. (4.15) or Eqs. (4.23) and Eq. (4.30), form a coupled system of $3(N+1)$ ordinary differential equations for the solution $h_1, \ldots, h_{N+1}$, $\Gamma_1, \ldots, \Gamma_{N+1}$ and the mesh $x_1, \ldots, x_{N+1}$ with initial condition given by Eqs. (4.4,4.32). We use a staggered system for numbering the unknowns, i.e.,

$h_1, \Gamma_1, x_1, h_2, \Gamma_2, x_2, \ldots, h_{N+1}, \Gamma_{N+1}, x_{N+1},$

which provides a smaller bandwidth for the Jacobian matrix (note: the bandwidth=13 for the semi-discretisation in Eq. (4.15); it could be larger if smoothing of monitor function is used. This is in comparison to, for example, the numbering

$h_1, h_2, \ldots, h_{N+1}, \Gamma_1, \Gamma_2, \ldots, \Gamma_{N+1}, x_1, x_2, \ldots, x_{N+1},$

which although sparse has a much bigger bandwidth. This significantly improves the performance of the solver. We note here that the stiff solver DASPK [12] is used which utilises an iterative scheme based on Krylov subspace methods for the solution of the linearised system including preconditioning using Incomplete LU decomposition of the Jacobian matrix. This had a significant influence on the performance in comparison to DASSL. However, we need to choose a sufficiently large fill-in for the LU factorisation of the Jacobian, otherwise the convergence of the iterative solver is very slow.

In all the results presented below, the parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $L = 40$; the MMPDE 4 relaxation parameter $\tau = 10^{-3}$; the modified curvature-based monitor function parameters are:

$\beta = \omega = 10^5$ and the monitor function smoothing parameters are $p = 2$, $\gamma = 2$. We only show results based on the conservative semi-discretisation scheme Eq. (4.15). Our results using the non-conservative scheme Eq. (4.23) was not as accurate compared to the conservative scheme so we do not show these results here. Figures 4.4(a, b) show the time evolution of $h(x, t)$ and $\Gamma(x, t)$ respectively, to a travelling wave solution using the adaptive moving mesh scheme Eq. (4.15) with $N = 1000$ points corresponding to initial mesh size $\Delta x = 0.04$. The results are shown for t ranging between 0 to 90. We observe that the adaptive moving mesh scheme captures the essential features of the solution with less number of points ($N = 1000$ here compared to $N = 4 \times 10^4$ for the uniform mesh computation shown in figures 4.2(a, b). Figure 4.5(a, b) shows the corresponding equidistributing coordinate transformation $x = x(\xi, t)$ and trajectories. These figures show that the mesh points are redistributed with more points in the region near the leading edge ($\Delta x \approx 10^{-4}$), where there is a rapid variation in both h and Γ, and fewer points elsewhere ($\Delta x \approx 0.1$); note the initial $\Delta x = 0.04$ uniform everywhere. Figures 4.6(a, b) present the solution $h(x, t)$ and $\Gamma(x, t)$, respectively, at time $t = 90$. The insets show the increased number of points that are redistributed near the capillary ridge and capillary waves ahead and behind it, and the step-like structure (figure 4.6(a)) which are controlled by the curvature in h component of the monitor function. In addition, we see redistribution of points around the maximum in Γ and it's leading edge (figure 4.6(b)) controlled by the curvature in Γ component of the monitor function. In Figures 4.7(a, b) we compare the numerical solution for h and Γ, respectively, at $t = 90$, using the uniform mesh scheme Eqs.(4.5, 4.6) ($N = 10^3$ or $\Delta x = 0.04$ - solid blue line;

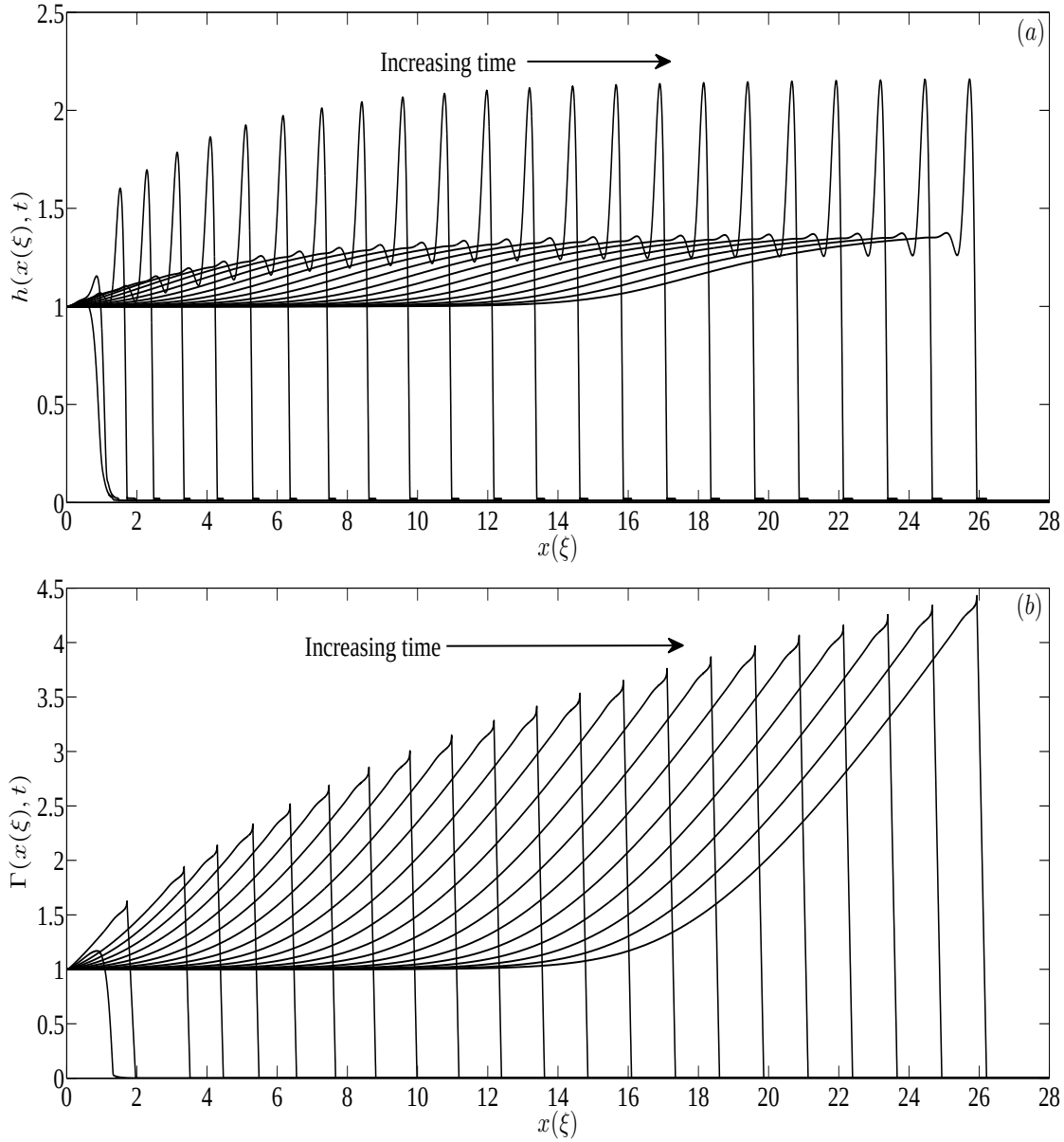

Figure 4.4: Time evolution of (a) $h(x,t)$ and (b) $\Gamma(x,t)$, using the adaptive moving mesh scheme given in Eq. (4.15). The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $L = 40$, $\Delta x = 0.04$ (so, $N = 10^3$ points), $\tau = 10^{-3}$, $\alpha = 1$, $\beta = \omega = 10^5$, $p = 2$ and $\gamma = 2$. The results are shown for time t ranging between 0 to 90.

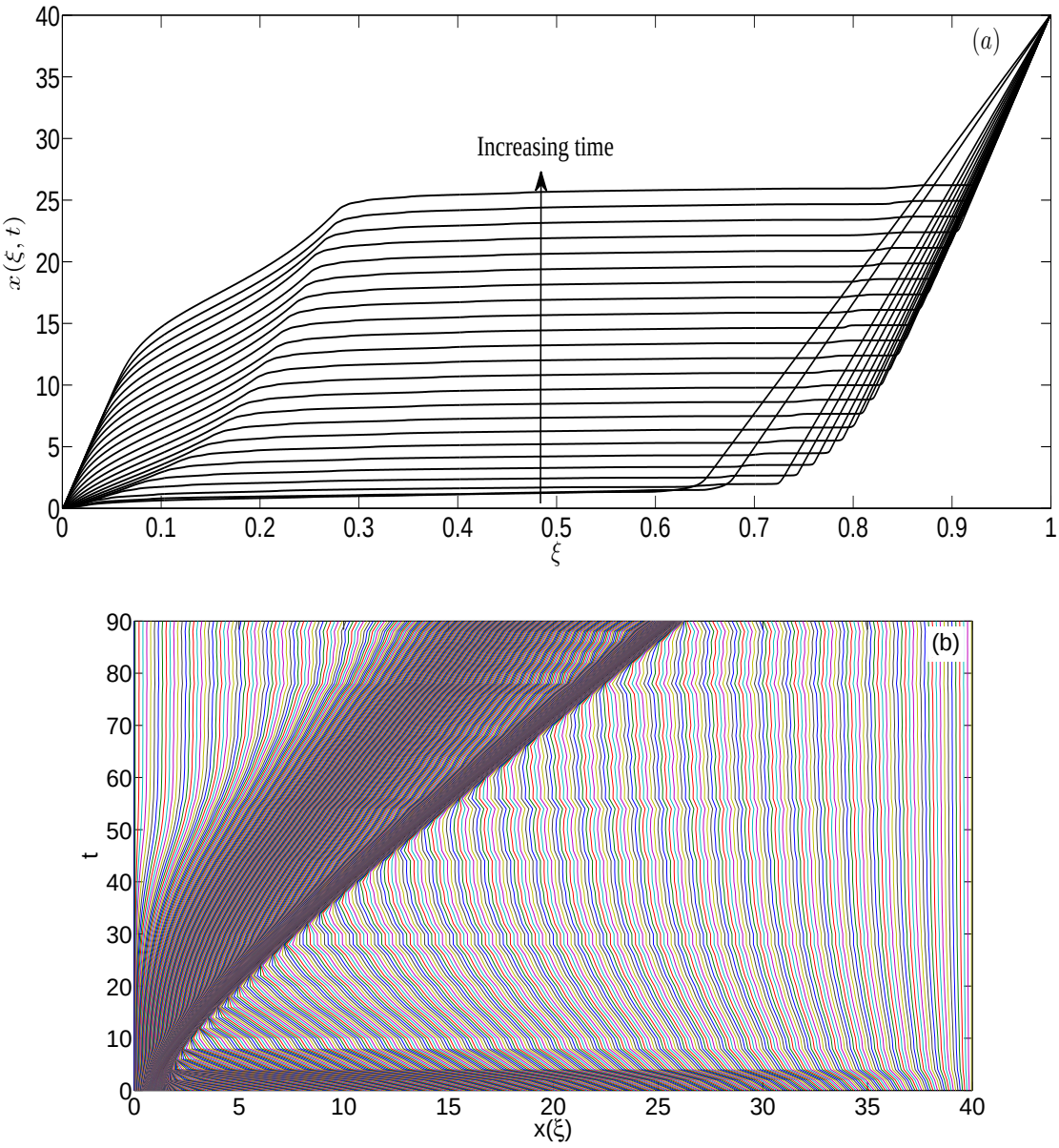

Figure 4.5: (a) The equidistributing coordinate transformations $x = x(\xi, t)$ and trajectories corresponding to the numerical solution shown in figure 4.4.

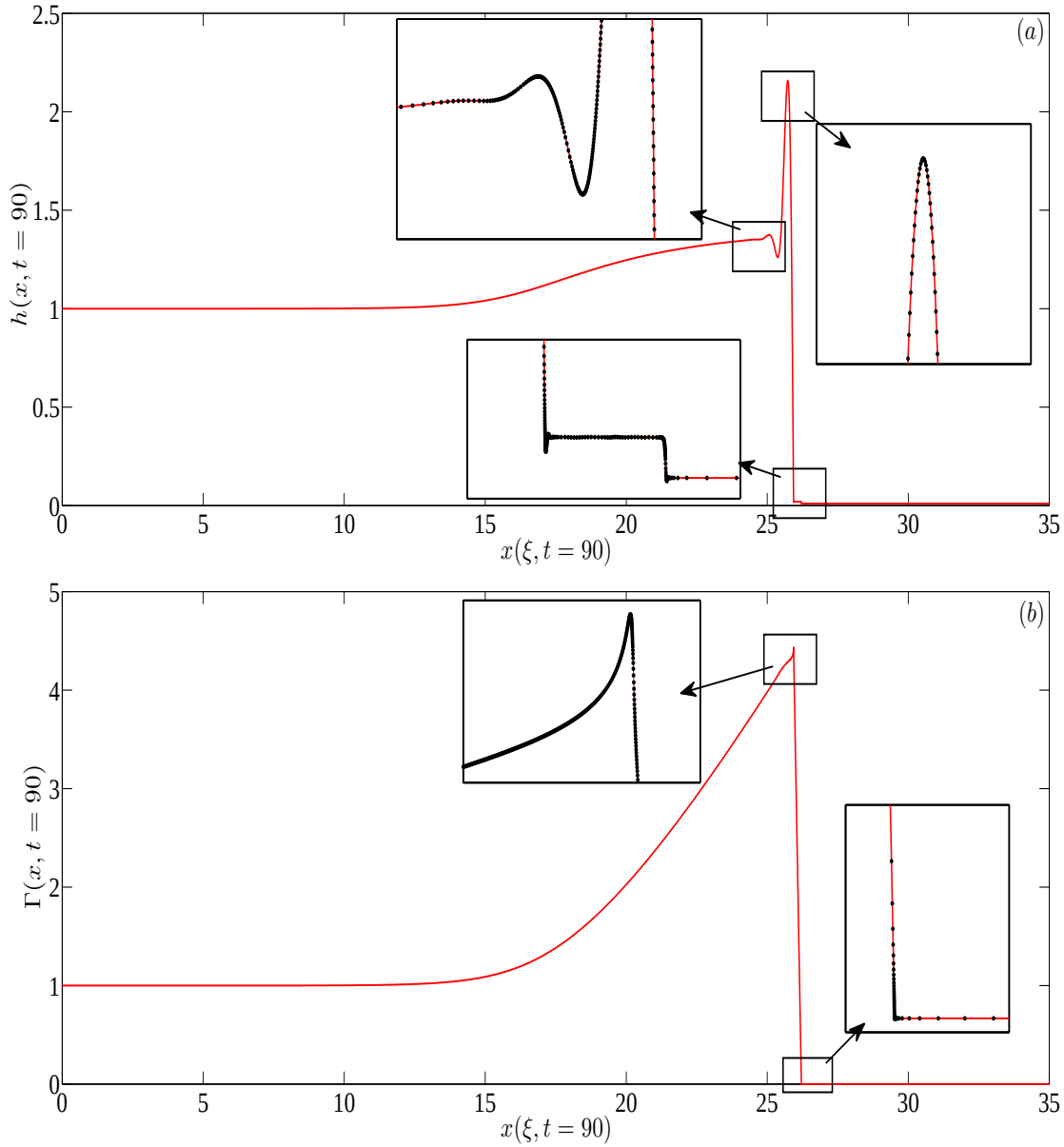

Figure 4.6: (a) $h(x, t = 90)$ and (b) $\Gamma(x, t = 90)$ obtained using the moving adaptive scheme with $N = 1000$ (initial $\Delta x = 0.04$). Insets in (a, b) show the zoomed-in wave-like structures in h and steep gradient in Γ around the location of it's maximum and leading edge. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $L = 40$, $\tau = 10^{-3}$, $\beta = \omega = 10^5$, $p = 2$ and $\gamma = 2$.

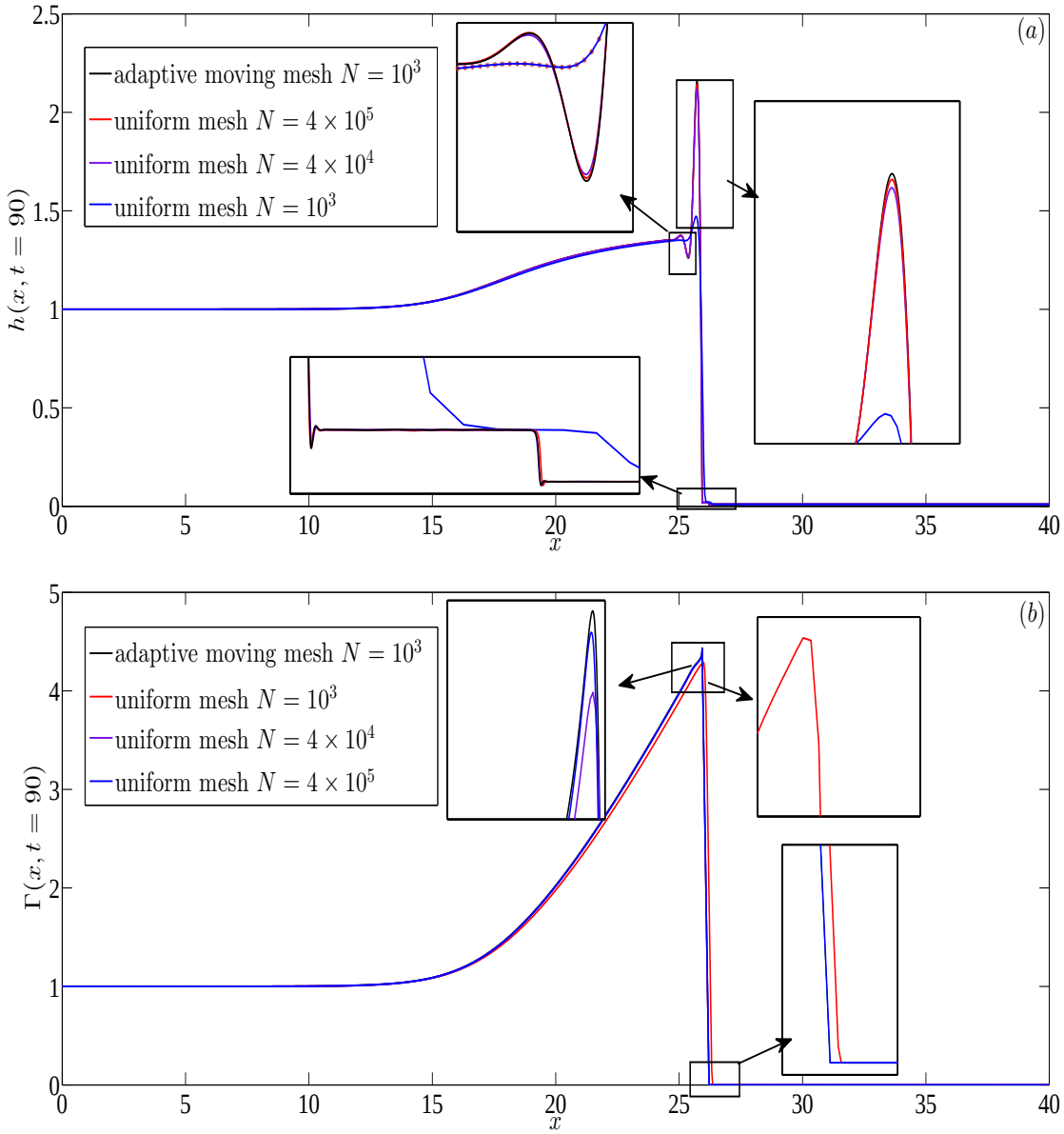

Figure 4.7: A comparison of the numerical solution of (a) $h(x, t = 90)$ and (b) $\Gamma(x, t = 90)$ using the uniform mesh scheme Eqs.(4.5, 4.6) ($N = 10^3$ or $\Delta x = 0.04$ - solid blue line; $N = 4 \times 10^4$ or $\Delta x = 10^{-3}$ - solid purple line; $N = 4 \times 10^5$ or $\Delta x = 10^{-4}$ - solid red line) and the adaptive moving mesh scheme Eq. (4.15) ($N = 1000$ or initial $\Delta x = 0.04$ - solid black line). The insets show the zoomed-in solution at locations where there is rapid variation in h and Γ. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $L = 40$, $\tau = 10^{-3}$, $\beta = \omega = 10^5$, $p = 2$ and $\gamma = 2$.

$N = 4 \times 10^4$ or $\Delta x = 10^{-3}$ - solid purple line; $N = 4 \times 10^5$ or $\Delta x = 10^{-4}$ - solid red line) and the adaptive moving mesh scheme Eq. (4.15) ($N = 1000$ or initial $\Delta x = 0.04$ - solid black line). We observe that the adaptive moving mesh numerical solution is almost indistinguishable from the uniform mesh numerical solution corresponding to $\Delta x = 10^{-3}, 10^{-4}$. Moreover, the insets in figures $4.7(a, b)$ show that there is still an error (albeit small) between the uniform mesh solution for $\Delta x = 10^{-3}, 10^{-4}$, particularly at the capillary ridge in h and the maximum in Γ. This implies that the accuracy and convergence of the uniform mesh scheme is restricted to very small values of Δx in these regions. In contrast, the adaptive moving mesh scheme by redistributing more points to these regions (starting from a modest number of points, $N = 1000$) can very effectively control the accuracy and convergence of the numerical solution there.

4.2 PDEs with constant volume boundary conditions

In this section we investigate the numerical solution of a spreading one-dimensional drop laden with surfactant using both a uniform mesh as well as an adaptive moving mesh. The underlying PDEs are given by Eqs. (4.1, 4.2). These PDEs are supplemented by boundary conditions, which are,

$$h = b, h_{xxx} = 0, \Gamma = 0, \quad \text{at} \quad x = \pm L, \tag{4.34}$$

where $\pm L$ are the boundaries of the physical domain and $b \ll 1$, is the precursor film thickness. The initial conditions are taken as (following [35, 37, 39]):

$$h(x,0) = (1-x^2)\left[H(x_0-x) - H(-x_0-x)\right] + b\left[H(x-x_0) + H(-x_0-x)\right], \quad (4.35)$$

$$\Gamma(x,0) = H(x_0-x) - H(-x-x_0), \quad (4.36)$$

where $H(x)$ is the Heaviside function and x_0, is the initial location where the parabolic-shaped drop connects to the precursor film both upstream and downstream. The initial condition for Γ assumes a uniform surfactant concentration between $\pm x_0$ and zero surfactant concentration ahead and behind it and is characterised by a steep gradient at x_0. The initial conditions are shown in figure 4.8 with $b = 0.01$, $x_0 = 1$ and $L = 2$.

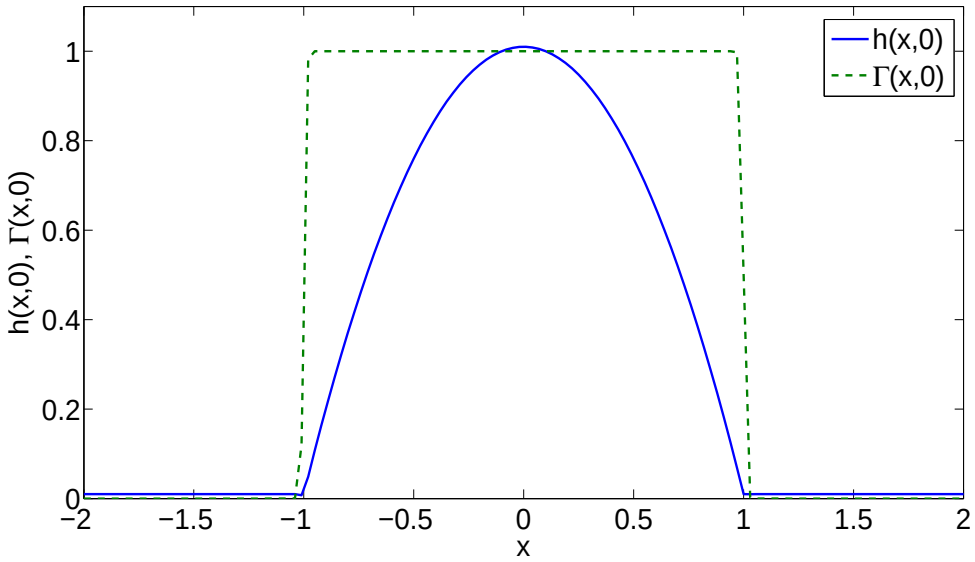

Figure 4.8: The initial condition for h and Γ with $b = 0.01$, $x_0 = 1$ and $L = 2$.

We seek the numerical solution of Eqs. (4.1, 4.2) subject to the BCs given by Eq.

(4.34) for the time evolution of the initial drop and surfactant concentration. We first show this solution using a uniform mesh followed by an adaptive moving mesh. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$ and $\delta = 10^{-5}$. The upstream and downstream domain length is fixed at $L_1 = -3$ and $L_2 = 17$, respectively. In all numerical simulations show we have ensured that the initial volume of fluid and surfactant are always conserved. Figures 4.9(a, b) illustrate the time evolution of h and Γ, respectively, using the above uniform mesh scheme with $N = 2\times10^4$ points (so, $\Delta x = 10^{-3}$). The results are shown for time t ranging between 0 to 400. In addition to the wave-like structures observed at the leading edge of the drop (similar to the constant flux case), there is now also a front in h travelling upstream of the trailing edge of the drop (see figure 4.9(a)). There is also a steep gradient in Γ near the trailing edge of the drop (see figure 4.9(b)) in addition to the one near the drop's leading edge. Thus, accurate resolution of the numerical solution for both h and Γ at multiple locations is required. This makes the numerical solution on a uniform mesh computationally intensive and hence the use of an adaptive moving mesh would be beneficial in this respect.

Figures 4.10(a, b, c) show the time evolution of $h(x, t)$, $\Gamma(x, t)$ and $x = x(\xi, t)$, respectively, using the adaptive moving mesh scheme Eq. (4.15) with $N = 1500$ points corresponding to initial mesh size $\Delta\xi = 0.013$. The results are shown for t ranging between 0 to 400. We observe that the adaptive moving mesh scheme captures the essential features of the solution with less number of points ($N = 1500$ here compared

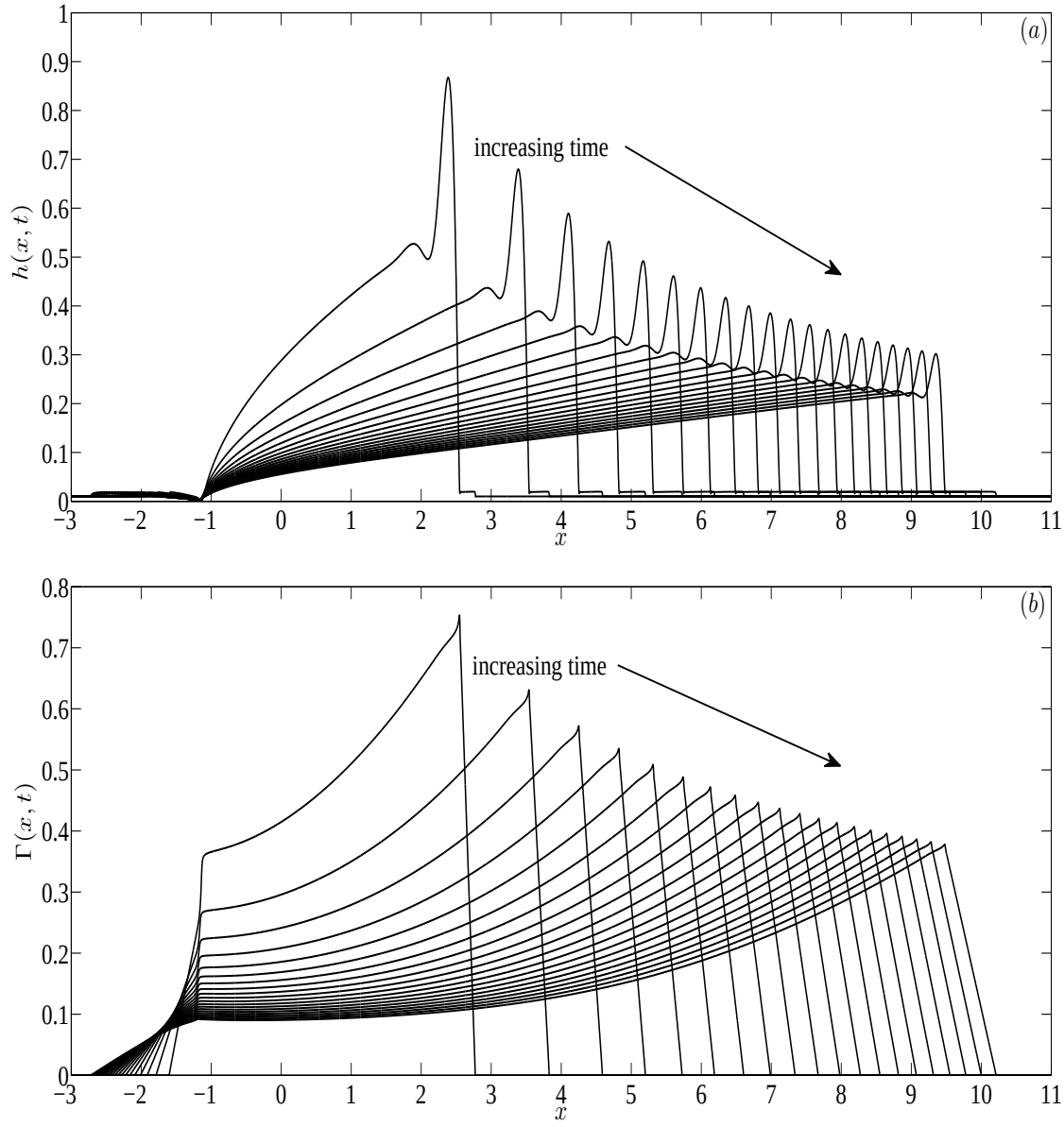

Figure 4.9: Time evolution of (a) $h(x,t)$ and (b) $\Gamma(x,t)$, using the uniform mesh scheme given in Eqs. (4.5,4.6) with $N = 2 \times 10^4$ points (so, $\Delta x = 10^{-3}$) and $x \in (-3, 17)$. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$ and $\delta = 10^{-5}$. The results are shown for time t ranging between 0 to 400.

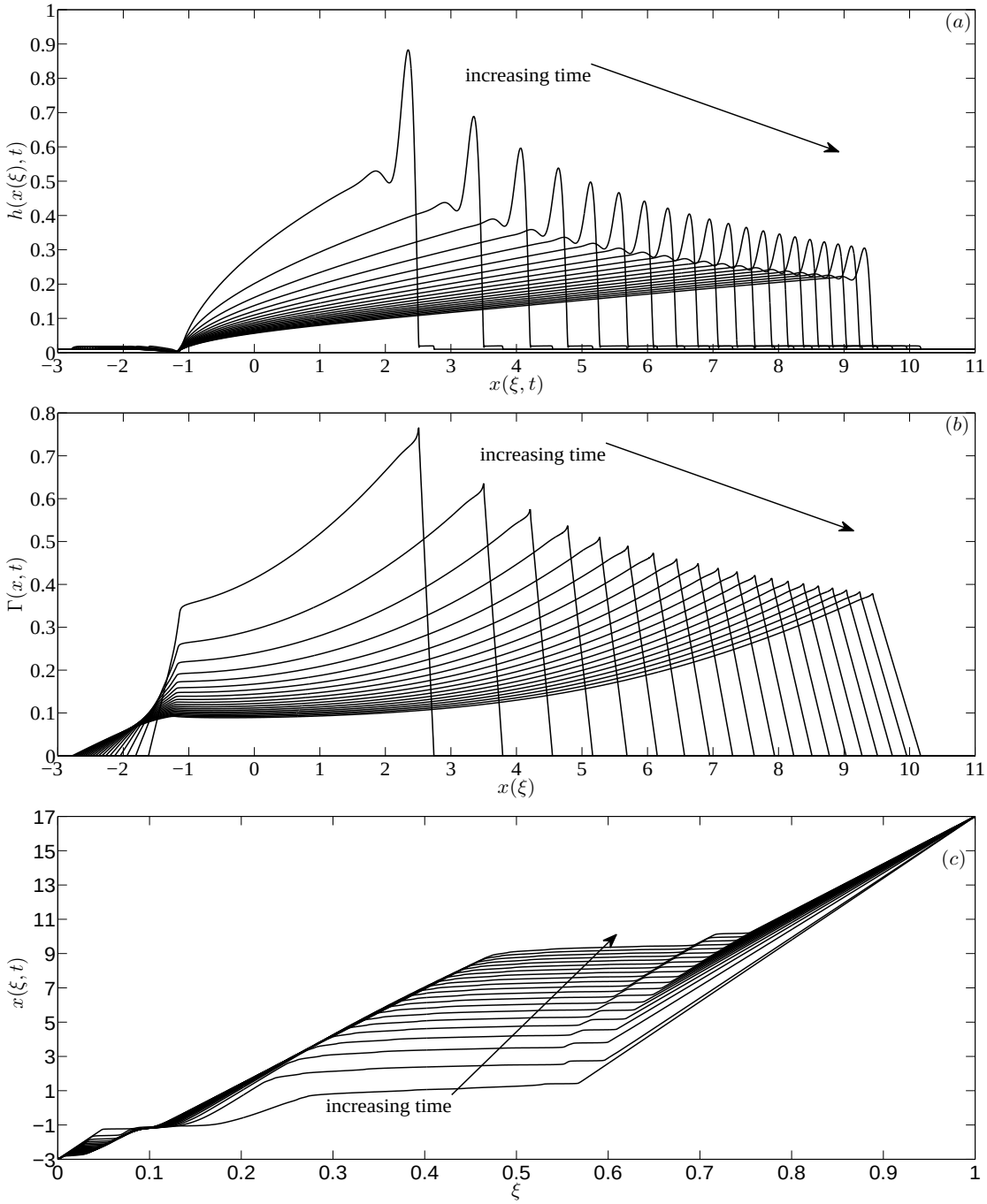

Figure 4.10: Time evolution of (a) $h(x,t)$, (b) $\Gamma(x,t)$ and (c) $x = x(\xi,t)$, using the adaptive moving mesh scheme given in Eq. (4.15) with $N = 1500$ (initial $\Delta x = 0.013$) and $x \in (-3, 17)$. The results are shown for time t ranging between 0 to 400. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $\tau = 10^{-3}$, $\beta = \omega = 10^5$, $p = 2$ and $\gamma = 2$.

to $N = 2 \times 10^4$ for the uniform mesh computation shown in figures $4.9(a, b)$). The corresponding equidistributing coordinate transformation $x = x(\xi, t)$ in figure $4.10(c)$ shows that the mesh points are redistributed with more points in the region near the leading edge and trailing edges ($\Delta x \approx 10^{-4}$), where there is a rapid variation in both h and Γ, and fewer points elsewhere ($\Delta x \approx 0.1$); note the initial $\Delta x = 0.013$ uniform everywhere.

Figures $4.11(a, b)$ present the solution $h(x, t)$ and $\Gamma(x, t)$, respectively, at time $t = 400$. The insets show the increased number of points that are redistributed near the capillary ridge and capillary waves ahead and behind it, the step-like structure ahead of the drop's leading edge and the front upstream of the drop's trailing edge (see figure $4.11(a)$) which are controlled by the curvature in h component of the monitor function. In addition, we see redistribution of points around the maximum in Γ and it's leading edge and trailing edge (see figure $4.11(b)$) controlled by the curvature in Γ component of the monitor function.

In Figures $4.12(a, b)$ we compare the numerical solution for h and Γ, respectively, at $t = 90$, using the uniform mesh scheme Eqs.(4.5, 4.6) ($N = 1500$ or $\Delta x = 0.013$ - solid black line; $N = 2 \times 10^4$ or $\Delta x = 10^{-3}$ - solid blue line) and the adaptive moving mesh scheme Eq. (4.15) ($N = 1500$ or $\Delta\xi = 0.013 =$ initial Δx - solid red line). We observe that the adaptive moving mesh numerical solution is almost indistinguishable from the uniform mesh numerical solution corresponding to $\Delta x = 10^{-3}$. However, on closer inspection there are discernible differences between the two solutions for both h

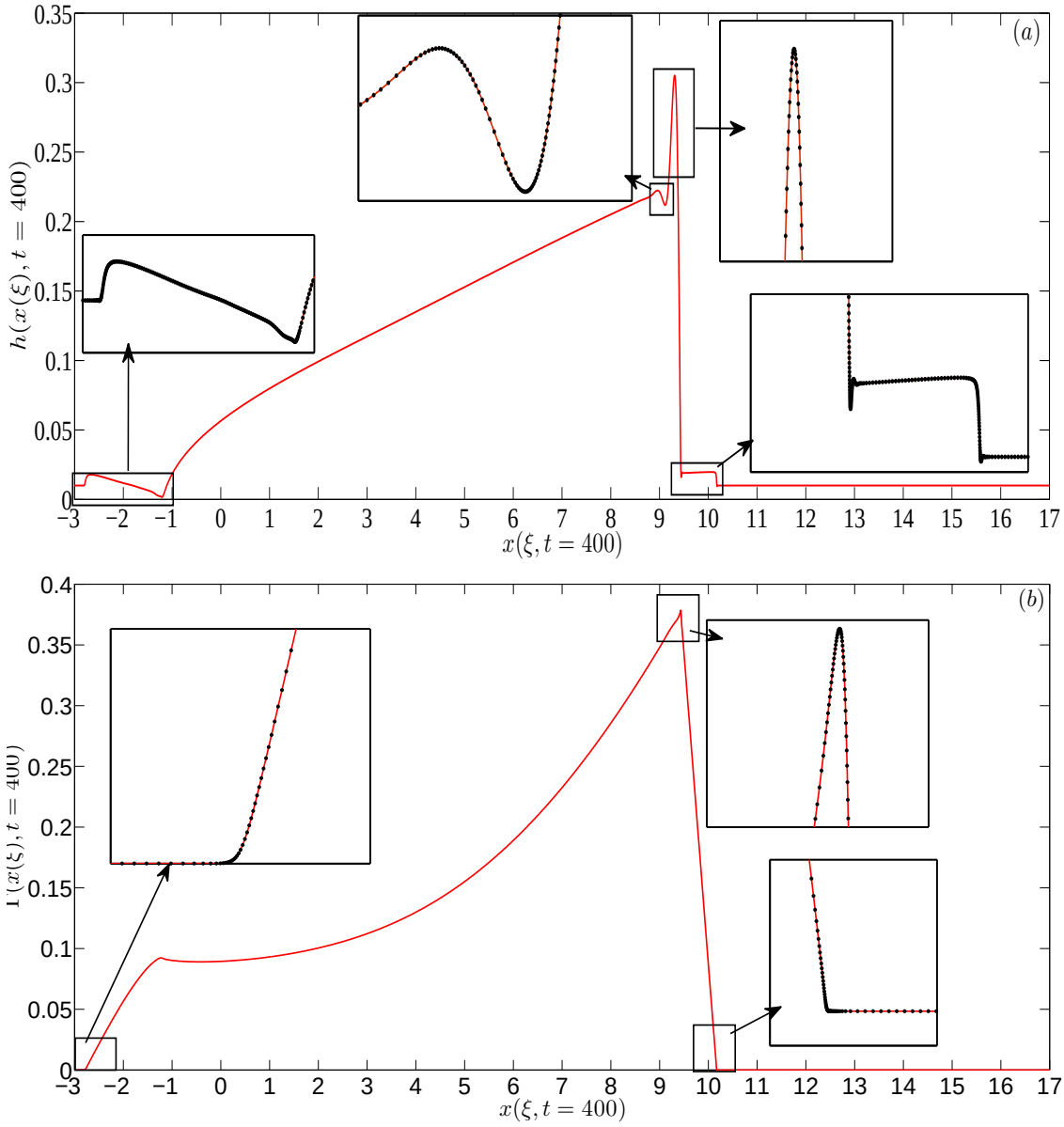

Figure 4.11: (a) $h(x, t = 400)$ and (b) $\Gamma(x, t = 400)$ obtained using the moving adaptive scheme with $N = 1500$ ($\Delta\xi = 0.013 =$ initial Δx). Insets in (a, b) show the zoomed-in wave-like structures in h and steep gradient in Γ around the drop's leading and trailing edges. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $\tau = 10^{-3}$, $\beta = \omega = 10^5$, $p = 2$ and $\gamma = 2$.

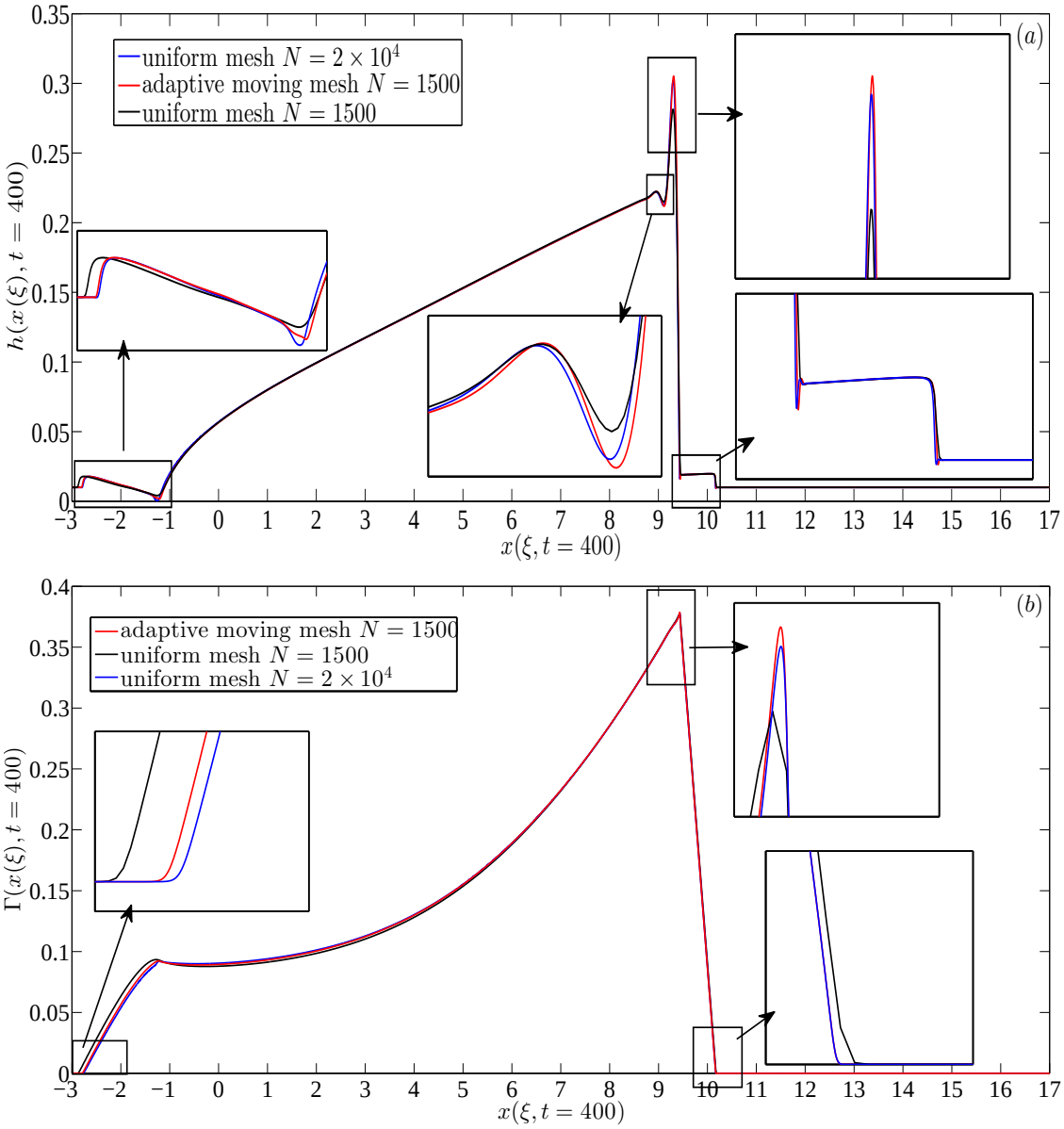

Figure 4.12: A comparison of the numerical solution of (a) $h(x, t = 400)$ and (b) $\Gamma(x, t = 400)$ using the uniform mesh scheme Eqs.(4.5, 4.6) ($N = 1500$ or $\Delta x = 0.013$ - solid black line; $N = 2 \times 10^4$ or $\Delta x = 10^{-3}$ - solid blue line) and the adaptive moving mesh scheme Eq. (4.15) ($N = 1500$ or $\Delta\xi = 0.013 =$ initial Δx - solid red line). The insets show the zoomed-in solution at locations where there is rapid variation in h and Γ. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$, $G = 1$, (so, $\alpha_1 = 1$, $\beta_1 = 0$), $b = 10^{-2}$, $\delta = 10^{-5}$, $\tau = 10^{-3}$, $\beta = \omega = 10^5$, $p = 2$ and $\gamma = 2$.

and Γ, particularly, near the leading and trailing edges of the drop (see solid blue and red lines in the insets in figures 4.12(a, b)). We believe that the uniform mesh solution at $\Delta x = 10^{-3}$ has not fully converged in these regions in comparison to the adaptive mesh solution where $\Delta x \approx 10^{-4}$ in these regions. We would need to use a smaller value of Δx to confirm this which we have not done here. This again highlights the usefulness of the adaptive moving mesh scheme in redistributing more points to these regions (starting from a modest number of points, $N = 1500$) enabling to effectively control the accuracy and convergence of the numerical solution there.

4.3 Conclusions

We have successfully implemented a moving adaptive mesh scheme based on moving mesh PDEs (MMPDEs) for a coupled system of two parabolic PDEs. The main highlight of the results is in adapting the monitor function to include multiple solution components (based on Huang & Russell [53]). This enables us to accurately resolve the complicated multiple wave-like structures (which need not necessarily be aligned for both solution components)) compared to the uniform mesh scheme (see figures 4.7, 4.12).

In conclusion, our results indicate great promise for MMPDEs in thin-film flow problems which involving couple evolutionary PDEs. Although we have only considered a specific form of the underlying PDEs, the general framework presented in this chapter

can be utilised for a thin-film equation coupled to other PDEs such as, electric field, temperature (in non-isothermal problems), etc., of the form considered here.

Chapter 5

Numerical Solution of a Two-dimensional Fourth Order Parabolic PDE on a Moving Adaptive Mesh

5.1 PDE with constant flux boundary conditions

Consider the two-dimensional fourth order degenerate nonlinear parabolic PDE described in Chapter 1,

$$h_t + \boldsymbol{\nabla}\cdot\left[\frac{1}{3}Ca\ h^3\boldsymbol{\nabla}\nabla^2 h - \frac{1}{3}D(\theta)h^3\boldsymbol{\nabla}h\right] + \left[\frac{1}{3}h^3\right]_x = 0, \qquad (x,y,t)\in R^2\times[0,T_f]. \quad (5.1)$$

Here, $\boldsymbol{\nabla} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)^T$ is the gradient operator and $\nabla^2 = \left(\frac{\partial^2}{\partial x^2}, \frac{\partial^2}{\partial y^2}\right)$ is the Laplacian operator. The dimensionless parameters are the Capillary number Ca (related to the surface tension term), $D(\theta)$ is related to the vertical component of gravity and θ is the inclination angle. We consider a rectangular physical domain: $0 \leq x \leq L_x$ and $-L_y/2 \leq y \leq L_y/2$, where L_x and L_y are the lengths of the domain in the x and y directions, respectively and T_f is a specific time.

Equation (5.1) is supplemented by eight boundary conditions (BCs) in x and y. These are given by

$$h(0,y,t) = 1,\ h(L_x,y,t) = b,\ h_{xxx}(0,y,t) = h_{xxx}(L_x,y,t) = 0,\ \forall y \in [-L_y/2, L_y/2], \tag{5.2}$$

$$h_y(x,-L_y/2,t) = h_y(x,L_y/2,t) = h_{yyy}(x,-L_y/2,t) = h_{yyy}(x,L_y/2,t) = 0,\ \forall x \in [0, L_x], \tag{5.3}$$

Here, $b \ll 1$, is the precursor film thickness. This represents a constant flux of fluid introduced at the upstream end $x = 0$ and the substrate is pre-wetted with a thin precursor film of thickness b. We impose symmetry BCs at the y boundaries.

The main aim here is to obtain numerical solutions that demonstrate the fingering instability similar to that observed in figure 1.1 in Chapter 1. Linear stability analysis of this problem using the one-dimensional travelling wave solution obtained from Eq. (3.1) as the base state have shown this solution to be linearly unstable to small-amplitude sinusoidal perturbations in the transverse y-direction (see Bertozzi & Brenner [7] and

Kondic & Diez [58]).

The numerical dispersion relation computed by Bertozzi and Brenner [7] showed that the longer wavelengths are linearly unstable with a well-defined maximum wavelength and corresponding growth rate. The shorter wavelengths are stabilised by capillary or surface tension effects.They performed a small wavenumber analysis to obtain a stability criterion which showed that the capillary ridge is necessary for the long wavelength instability. They also showed that there is a critical angle above the horizontal beyond which the spreading flow is always linearly unstable with the maximum growth rate and band of unstable wavenumbers increasing. This critical angle is implicitly a function of the system parameters Ca and D. Two-dimensional numerical studies of this problem analysing the nonlinear evolution of the transverse instability have been done by several authors, most notably by Kondic and Diez [58]. They showed that by varying the inclination angle the shape of the fingering instabilities and the surface coverage varied considerably. Decreasing the angle of inclination resulted in saw-tooth, or triangular-type patterns being observed. As the angle of inclination was increased the fingers were more rounded with parallel sides. This gave quantitative agreement to the corresponding experimental work. These numerical studies were performed using finite differences on a uniform mesh. Our goal is to accurately capture the fingering behaviour also including adaptation of the mesh in the transverse y-direction along with that in the x-direction described in the earlier chapters. Based on this, we superpose sinusoidal perturbations in the y direction locally on the one-dimensional

travelling wave solution. Using this superposed initial condition for Eq. (5.1,5.2,5.3) we then track the evolution of the two-dimensional solution to determine the nonlinear fingering behaviour.

The superposed initial condition is chosen as

$$h(x,y,0) = h_0(x) + \sum_{l=1,k=1}^{m,n} a_k \cos(k\pi y)\ e^{\left[-K_l\ (x-x_l)^2\right]}. \tag{5.4}$$

where, $h_0(x)$ is the travelling wave solution, k is the wavenumber of each sinusoidal mode with period $= 2/k$ and amplitude a_k, n is the total number of modes imposed, m is the total number of locations $x = x_l$ in the x direction across where the sinusoidal perturbations are applied and K_l controls the width of the localised perturbations at x_l. Figure 5.1 shows the two-dimensional initial condition Eq. (5.4) with a single transverse perturbation ($k = n = 1$) applied at two locations: x_1, location where h_0 has a maximum (i.e., at the capillary ridge) and x_2, the location of the effective contact line where the front connects onto the precursor film. The corresponding amplitude of the perturbation at each location is $a_1 = 0.1$ and $a_2 = 0.01$, respectively. We choose $K_1 = 20$, $K_2 = 100$, the precursor thickness $b = 0.1$, $L_x = 20$ and $L_y = 2$. The travelling wave solution $h_0(x)$ is obtained from the solution of Eq. (3.1) at $t = 2$ (by this time the one-dimensional solution has already developed into a travelling wave solution). The choice of the above parameters are informed from previous studies, for example Kondic & Diez [58].

We now seek the numerical solution of Eq. (5.1) with the above boundary and initial

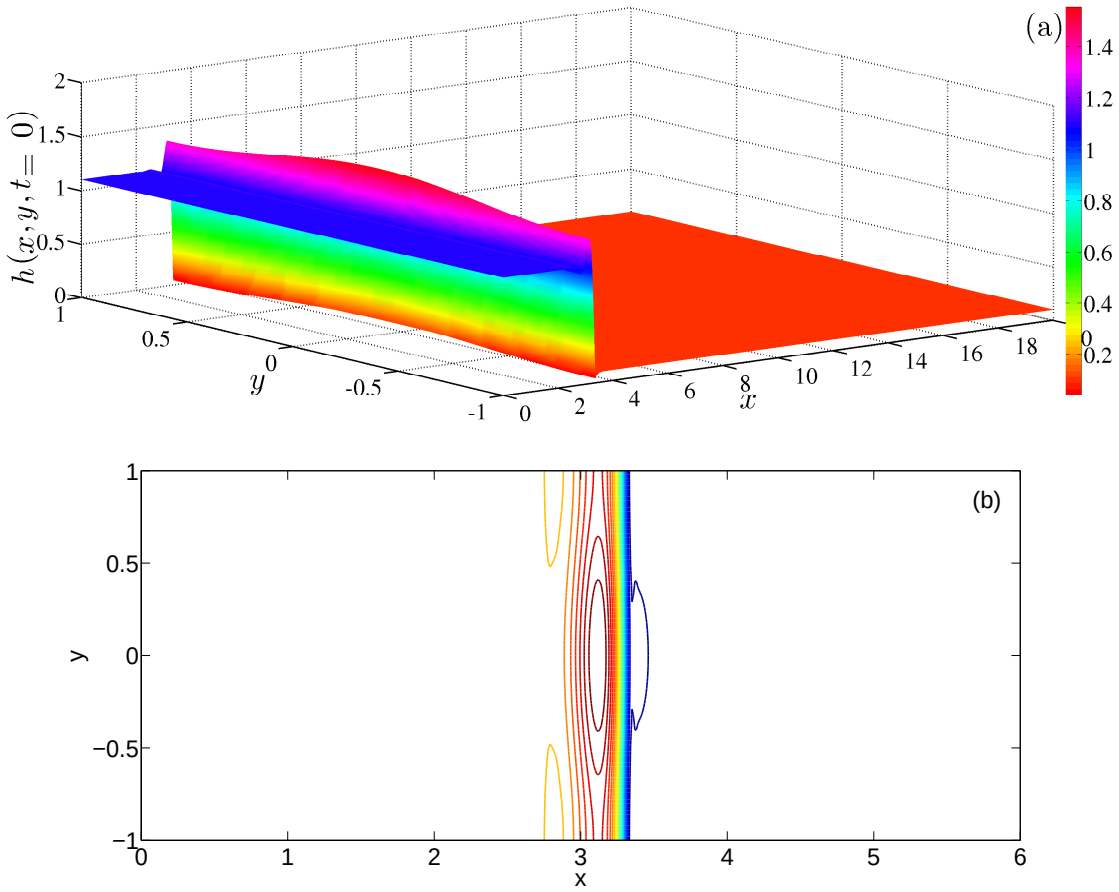

Figure 5.1: (a) Surface and (b) contour plots showing the two-dimensional initial condition given by Eq. (5.4) with $k = n = 1$, $m = 2$, $a_1 = 0.1$, $a_2 = 0.01$, $K_1 = 20$, $K_2 = 100$, $b = 0.1$, $L_x = 20$ and $L_y = 2$. x_1 is the location where h_0 has a maximum (i.e., at the capillary ridge) and x_2 is the location of the effective contact line where the front connects onto the precursor film. The travelling wave solution $h_0(x)$ is obtained from the solution of Eq. (3.1) at $t = 2$.

condition to determine the evolution of the film thickness $h(x, y, t)$ using finite difference methods. The numerical method is performed first on a uniform mesh and then on an adaptive moving mesh.

5.1.1 Numerical solution of Eqs. (5.1,5.2,5.3) on a uniform mesh

The domain $[0, L_x]\times[-L_y/2, L_y/2]$ is uniformly divided into $(M_x+1)\times(N_y+1)$ discrete points, respectively, as follows:

$(x_j, y_k) = ((j-1)\Delta x, (k-1)\Delta y - L_y/2), \quad j = 1, 2, \ldots, M_x+1, k = 1, 2, \ldots, N_y+1,$

where $(\Delta x, \Delta y) = (\frac{L_x}{M_x}, \frac{L_y}{N_y})$ is the width of each sub-interval. We discretise the spatial derivatives appearing in Eq. (5.1) using finite differences, keeping the time derivative continuous. The spatial discretisation is done as follows. We define a forward and backward finite difference as

$$(h_x)_{j+\frac{1}{2},k} = \frac{h_{j+1,k} - h_{j,k}}{\Delta x} + O(\Delta x), \qquad (h_{\bar{x}})_{j-\frac{1}{2},k} = \frac{h_{j-1,k} - h_{j,k}}{\Delta x} + O(\Delta x), \qquad (5.5)$$

$$(h_y)_{j,k+\frac{1}{2}} = \frac{h_{j,k+1} - h_{j,k}}{\Delta y} + O(\Delta y), \qquad (h_{\bar{y}})_{j,k-\frac{1}{2}} = \frac{h_{j,k-1} - h_{j,k}}{\Delta y} + O(\Delta y). \qquad (5.6)$$

Thus, a semi-discretisation of Eq. (5.1), keeping the time derivative continuous, can be written as

$$(h_t)_{j,k} + \frac{1}{3}Ca\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}\nabla^2 h\right]_{j,k} - \frac{1}{3}D(\theta)\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}h\right]_{j,k} + \frac{1}{3}(h^3)_{x_{j,k}} = 0. \qquad (5.7)$$

The surface tension term can be discretised in the following two ways:

1. Using Eqs. (5.5,5.6) we can write

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}\nabla^2 h\right]_{,j,k} = \frac{1}{\Delta x^2}\left[\left(h^3\left(h_{x\bar{x}x}+h_{y\bar{y}x}\right)\right)_{j+\frac{1}{2},k} - \left(h^3\left(h_{x\bar{x}x}+h_{y\bar{y}x}\right)\right)_{j-\frac{1}{2},k}\right] + \frac{1}{\Delta y^2}\left[\left(h^3\left(h_{x\bar{x}y}+h_{y\bar{y}y}\right)\right)_{j,k+\frac{1}{2}} - \left(h^3\left(h_{x\bar{x}y}+h_{y\bar{y}y}\right)\right)_{j,k-\frac{1}{2}}\right]. \quad (5.8)$$

2. Writing $\nabla^2 h(x,y,t) = G(x,y,t)$, we can discretise

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}G\right]_{j,k} = \frac{1}{\Delta x^2}\left[h^3_{j+\frac{1}{2},k}\left(G_{j+1,k}-G_{j,k}\right) - h^3_{j-\frac{1}{2},k}\left(G_{j,k}-G_{j-1,k}\right)\right] + \frac{1}{\Delta y^2}\left[h^3_{j,k+\frac{1}{2}}\left(G_{j,k+1}-G_{j,k}\right) - h^3_{j,k-\frac{1}{2}}\left(h_{j,k}-h_{j,k-1}\right)\right], \quad (5.9)$$

where

$$G_{j,k} = \frac{1}{\Delta x^2}\left[h_{j+1,k}-2h_{j,k}+h_{j-1,k}\right] + \frac{1}{\Delta y^2}\left[h_{j,k+1}-2h_{j,k}+h_{j,k-1}\right)]. \quad (5.10)$$

The vertical component of gravity term is discretised as

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}h\right]_{j,k} = \frac{1}{\Delta x^2}\left[\left(h^3\right)_{j+\frac{1}{2},k}\left(h_{j+1,k}-h_{j,k}\right) - \left(h^3\right)_{j-\frac{1}{2},k}\left(h_{j,k}-h_{j-1,k}\right)\right] + \frac{1}{\Delta y^2}\left[\left(h^3\right)_{j,k+\frac{1}{2}}\left(h_{j,k+1}-h_{j,k}\right) - \left(h^3\right)_{j,k-\frac{1}{2}}\left(h_{j,k}-h_{j,k-1}\right)\right]. \quad (5.11)$$

The horizontal component of gravity term is discretised as

$$(h^3)_{x_{j,k}} = \frac{1}{\Delta x}\left[(h^3)_{j+\frac{1}{2},k} - (h^3)_{j-\frac{1}{2},k}\right]. \tag{5.12}$$

The nonlinear diffusivity $h^3_{j+\frac{1}{2},k}$ and $h^3_{j,k+\frac{1}{2}}$ can be approximated by averaging and can be written as

$$(h^3)_{j+\frac{1}{2},k} = \frac{h^3_{j+1,k} + h^3_{j,k}}{2}, \;\; (h^3)_{j,k+\frac{1}{2}} = \frac{h^3_{j,k+1} + h^3_{j,k}}{2}, \tag{5.13}$$

or

$$(h^3)_{j+\frac{1}{2},k} = \left(\frac{h_{j+1,k} + h_{j,k}}{2}\right)^3, \;\; (h^3)_{j,k+\frac{1}{2}} = \left(\frac{h_{j,k+1} + h_{j,k}}{2}\right)^3, \tag{5.14}$$

with similar formulas for $h^3_{j-\frac{1}{2},k}$ and $h^3_{j,k-\frac{1}{2}}$. It has been shown that there are particular choices for the approximations of $h^3_{j+\frac{1}{2},k}$ and $h^3_{j,k+\frac{1}{2}}$ that have special properties, for example, the so-called positivity-preserving scheme, meaning that if one starts from strictly positive data for h, the scheme will help preserving this property. The interested reader is referred to the works by Kondic & Diez [30, 58] for details.

Evaluating the above at $j = 1, M_x + 1$ and $k = 1, N_y + 1$ require fictitious points $h_{0,k}$, $h_{M_x+2,k}$, $h_{j,0}$, $h_{j,-1}$, h_{j,N_y+2} and h_{j,N_y+3} which are obtained by discretising the boundary conditions Eqs. (5.2,5.3) using centred finite differences. Therefore, $h_{j,0} = h_{j,2}$, $h_{j,-1} = h_{j,3}$, $h_{j,N_y+2} = h_{j,N_y}$, $h_{j,N_y+3} = h_{j,N_y-1}$, $h_{0,k} = h_{2,k}$ and $h_{M_x+2,k} = h_{M_x,k}$. The boundary

conditions $h(x=0,y,t)=1$, $h(x=L_x,y,t)=b$ are replaced by thier ODE form:

$$h_{t,1,k}=0, \quad h_{t,M_x+1,k}=0, \quad k=1,\ldots,N_y+1. \tag{5.15}$$

The semi-discetisation scheme Eq. (5.7) is second order accurate and uses a 13-point stencil. It therefore requires a smaller bandwidth in comparison to a 25-point stencil if using a centred finite difference scheme in each direction.

Equations (5.7,5.15) form a system of $(M_x+1)\times(N_y+1)$ ordinary differential equations for the solution $h_{j,k}$ for $j=1,\ldots,M_x+1$ and $k=1,\ldots,N_y+1$ with initial condition given by Eq. (5.4). It is worth noting that although we have written the solution as $h_{j,k}$, for computational purposes it is better to write it as $h_{k,j}$. This is because we anticipate less number of mesh points in the y-direction compared to the x-direction (since the solution develops more structure in the x-direction requiring more mesh points for accurate resolution as seen in Chapter 3). Numbering the unknowns this way reduces the bandwidth of the system (albeit sparse) which significantly improves the performance of the solver. The resulting system of ODEs are solved using the stiff ODE solver DASPK [12]. This solver utilises an iterative scheme based on Krylov subspace methods for the solution of the linearised system including preconditioning using Incomplete LU decomposition of the Jacobian matrix. This had a significant influence on the performance in comparison to DASSL. However, we need to choose a sufficiently large fill-in for the LU factorisation of the Jacobian, otherwise the convergence of the iterative solver is very slow.

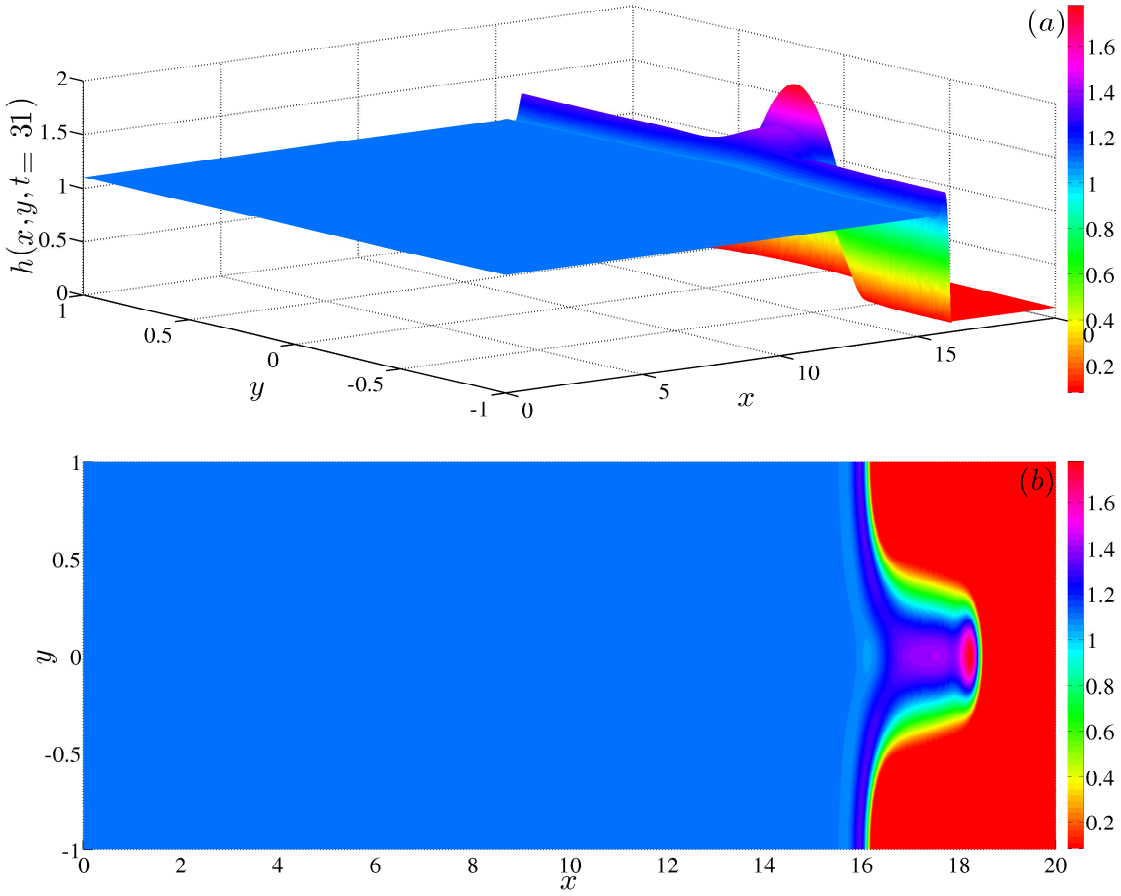

Figure 5.2: Surface plot ((a) side view and (b) top view) of $h(x, y, t)$ at $t = 31$. We clearly observe the formation of a fingering instability. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_x = 20$, $L_y = 2$, $M_x = 2000$ (so, $\Delta x = 0.01$) and $N_y = 200$ (so, $\Delta y = 0.01$).

In all the results presented below, the parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_x = 20$, $L_y = 2$, $M_x = 2000$ (so, $\Delta x = 0.01$) and $N_y = 200$ (so, $\Delta y = 0.01$). We have checked (but not shown here) that the above discretisation scheme converges for $(\Delta x, \Delta y) \to 0$ and the numerical solution shown below for $\Delta x = \Delta y = 0.01$ is accurate enough to be considered a converged solution. Figure 5.2(a, b) illustrate the surface plots (side and top view, respectively) of $h(x, y, t)$ at time $t = 31$. We clearly observe the formation of a single finger which is consistent with the linear stability analysis for the parameter values considered here. Figure 5.3(a, b, c) illustrate the surface plots of $h(x, y, t)$ showing its evolution in time (times shown are $t = 11$, $t = 20$ and $t = 31$). At early time $t = 11$, a finger is observed to slowly start forming. As time t increases, the finger appears to develop with a preferred width which is consistent with the linear stability analysis. This is more clearly depicted in the contour plots presented in figure 5.4(a, b, c) where the width of the finger is approximately one and it lengthens as time t increases.

5.1.2 Numerical solution of Eqs. (5.1,5.2,5.3) on an adaptive moving mesh

We now seek the numerical solution of Eqs. (5.1,5.2,5.3) using an adaptive moving mesh. In this section, we present the computational domain Ω_c, which is mapped to the physical domain Ω_p. We donate the computational and physical coordinates by

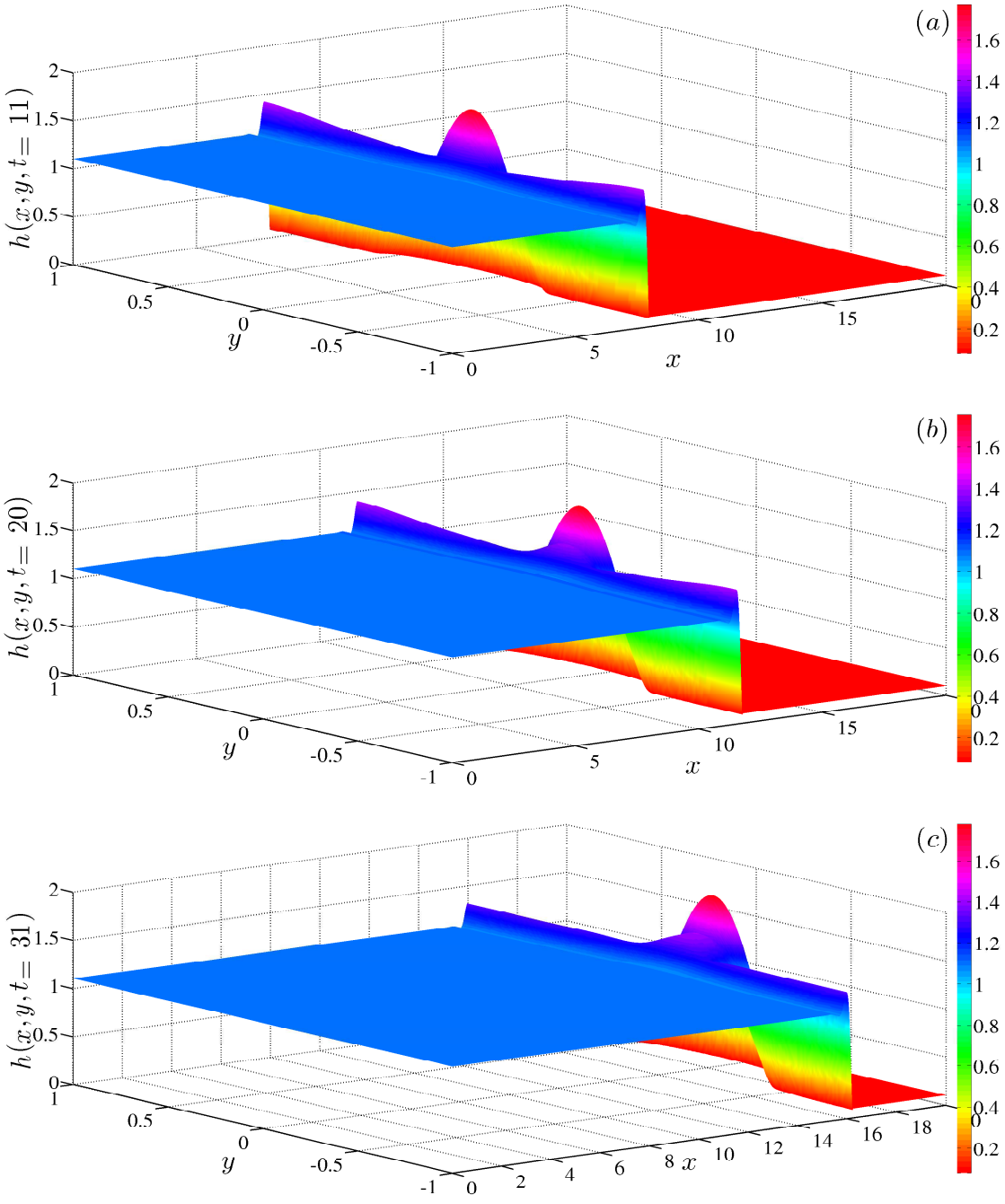

Figure 5.3: Surface plots of $h(x, y, t)$ showing its evolution in time (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_x = 20$, $L_y = 2$, $M_x = 2000$ (so, $\Delta x = 0.01$) and $N_y = 200$ (so, $\Delta y = 0.01$).

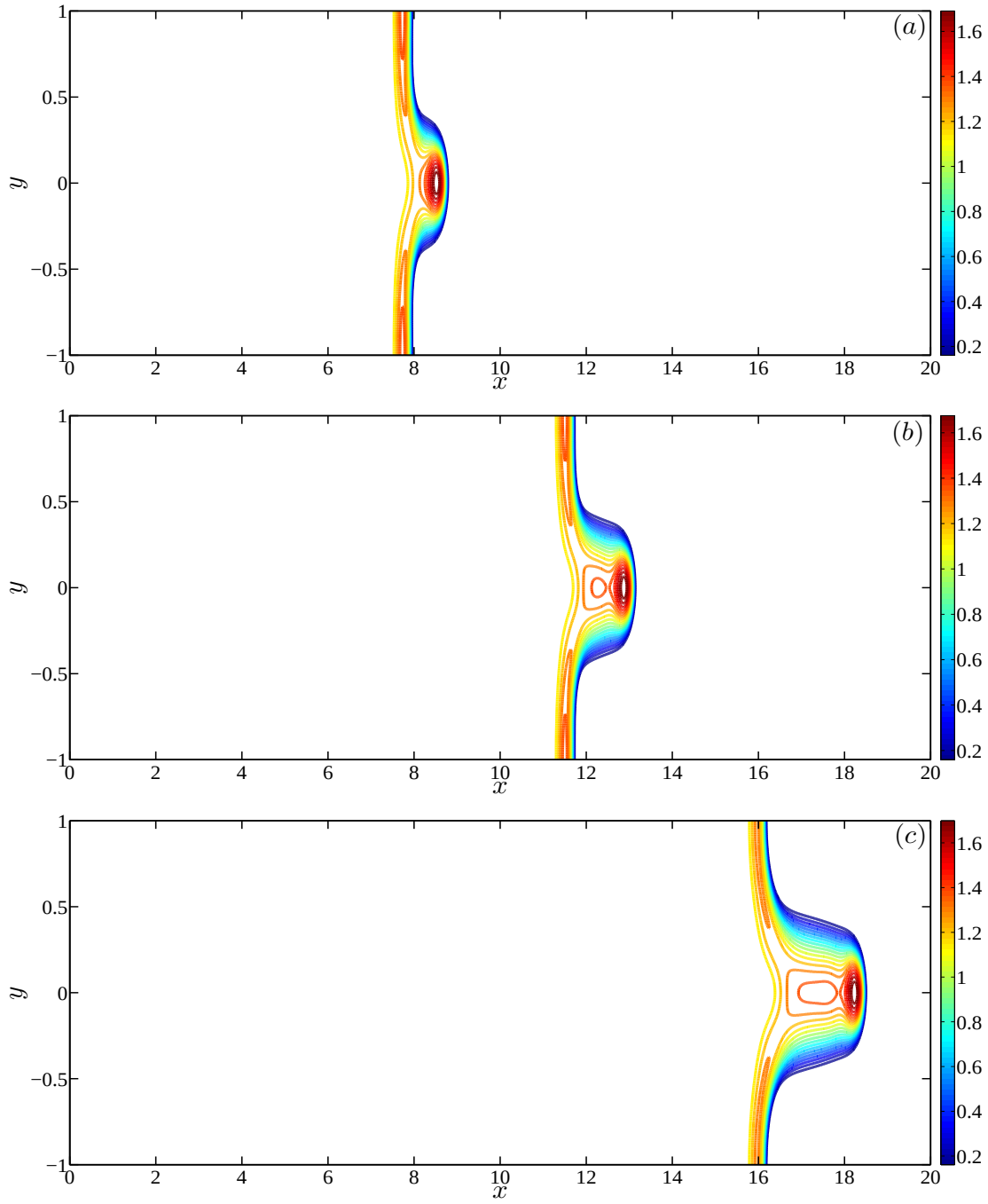

Figure 5.4: Contour plots of $h(x, y, t)$ showing its evolution in time (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_x = 20$, $L_y = 2$, $M_x = 2000$ (so, $\Delta x = 0.01$) and $N_y = 200$ (so, $\Delta y = 0.01$).

$\boldsymbol{\xi} = (\xi, \eta)$ and $\boldsymbol{x} = (x(\xi, \eta, t), y(\xi, \eta, t))$, respectively. This yields

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}) : \Omega_c \equiv [0,1] \times [0,1] \rightarrow \Omega_p \equiv [0, L_x] \times [-L_y/2, L_y/2].$$

Then the solution can be written as $h(\boldsymbol{x}, t) \approx h(\boldsymbol{x}(\boldsymbol{\xi}), t)$. A uniform mesh on the computational domain with the computational coordinates is depicted as

$$\mathcal{J}_h^c(t) \colon \xi_j = (j-1)\Delta\xi,\ \eta_k = (k-1)\Delta\eta,\ \text{for } j = 1, \ldots, M_x+1,\ k = 1, \ldots, N_y+1, \quad (5.16)$$

and a moving mesh associated with the solution $h(\boldsymbol{x}(\boldsymbol{\xi}), t)$ is described by

$$\mathcal{J}_h(t) \colon \boldsymbol{x}_{j,k}(\boldsymbol{\xi}) = \boldsymbol{x}(\xi_j, \eta_k, t),\ \text{for } j = 1, \ldots, M_x + 1,\ k = 1, \ldots, N_y + 1, \quad (5.17)$$

where $\Delta\xi = \dfrac{1}{M_x}$ and $\Delta\eta = \dfrac{1}{N_y}$ denote the uniform grid size in the computational domain and M_x, N_y are given positive integers denoting the number of mesh points in the x and y direction respectively.. Eq. (5.1) is reformulated in Lagrangian form. This is given by

$$h_t - (h_x x_t + h_y y_t) + \boldsymbol{\nabla} \cdot [Ca \frac{h^3}{3} \boldsymbol{\nabla} \boldsymbol{\nabla}^2 h - D(\theta)\ \frac{h^3}{3}\ \boldsymbol{\nabla} h] + \left(\frac{h^3}{3} \right)_x = 0, \quad (5.18)$$

The derivatives are transformed in terms of the computational coordinates by using the chain rule, so that

$$\begin{bmatrix} h_\xi \\ h_\eta \end{bmatrix} = \begin{bmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{bmatrix} \begin{bmatrix} h_x \\ h_y \end{bmatrix}.$$

Thus, we have

$$\begin{bmatrix} h_x \\ h_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} y_\eta & -y_\xi \\ -x_\eta & x_\xi \end{bmatrix} \begin{bmatrix} h_\xi \\ h_\eta \end{bmatrix}, \tag{5.19}$$

where, the Jacobian matrix $\boldsymbol{J}$ and its determinant are given by

$$\boldsymbol{J} = \frac{\partial(x,y)}{\partial(\xi,\eta)} = \begin{bmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{bmatrix}, \qquad J = \det(\boldsymbol{J}) = x_\xi y_\eta - x_\eta y_\xi.$$

The inverse Jacobian matrix is given by

$$\boldsymbol{J}^{-1} = \begin{bmatrix} \xi_x & \eta_x \\ \xi_y & \eta_y \end{bmatrix}.$$

Thus, we obtain the transformation relations as

$$\begin{bmatrix} \xi_x & \eta_x \\ \xi_y & \eta_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} y_\eta & -y_\xi \\ -x_\eta & x_\xi \end{bmatrix}, \tag{5.20}$$

Substituting into Eq. (5.19), leads to

$$\begin{bmatrix} h_x \\ \\ h_y \end{bmatrix} = \begin{bmatrix} \xi_x & \eta_x \\ \\ \xi_y & \eta_y \end{bmatrix} \begin{bmatrix} h_\xi \\ \\ h_\eta \end{bmatrix}. \tag{5.21}$$

Therefore, the first order derivatives are transformed as follows

$$h_x = \left[\xi_x h_\xi + \eta_x h_\eta\right], \quad h_y = \left[\xi_y h_\xi + \eta_y h_\eta\right], \quad \left[h^3\right]_x = \left[\xi_x h_\xi^3 + \eta_x h_\eta^3\right]. \tag{5.22}$$

Consequently, the vertical gravity and the surface tension terms are accordingly transformed using the fact that $h_{xx} = (h_x)_x$ and $h_{yy} = (h_y)_y$ as follows:

$$\begin{aligned}
&\boldsymbol{\nabla} \cdot \left[h^3 \boldsymbol{\nabla} h\right] = \frac{1}{J}\left[\left(h^3 R\right)_\xi + \left(h^3 S\right)_\eta\right], \\
&G = \boldsymbol{\nabla}^2 h = \frac{1}{J}\left[R_\xi + S_\eta\right], \\
&\boldsymbol{\nabla} \cdot \left[h^3 \boldsymbol{\nabla}\boldsymbol{\nabla}^2 h\right] = \boldsymbol{\nabla} \cdot \left[h^3 \boldsymbol{\nabla} G\right] = \frac{1}{J}\left[\left(h^3 R_1\right)_\xi + \left(h^3 S_1\right)_\eta\right], \\
&R = \left[a\ h_\xi + c\ h_\eta\right], \quad S = \left[q\ h_\eta + c\ h_\xi\right], \quad R_1 = \left[a\ G_\xi + c\ G_\eta\right], \quad S_1 = \left[q\ G_\eta + c\ G_\xi\right], \\
&a = \frac{1}{J}\left((J\xi_x)^2 + (J\xi_y)^2\right), \quad q = \frac{1}{J}\left((J\eta_x)^2 + (J\eta_y)^2\right), \quad c = \frac{1}{J}\left((J\xi_x)(J\eta_x) + (J\xi_y)(J\eta_y)\right).
\end{aligned} \tag{5.23}$$

2D Moving mesh PDE and monitor functions

We use the two-dimensional MMPDEs described in Chapter 2. These are

Regularised MMPDE5:

$$\tau \left(1 - \gamma_1 \boldsymbol{\nabla}_{\boldsymbol{\xi}}^2\right) \boldsymbol{x}_t = \boldsymbol{\nabla} \cdot \left(\rho(\boldsymbol{x}, t) \boldsymbol{\nabla} \boldsymbol{x}\right). \tag{5.24}$$

MMPDE4:

$$\tau \boldsymbol{\nabla}_{\boldsymbol{\xi}}^2 \left(\rho(\boldsymbol{x}, t) \boldsymbol{x}_t\right) = -\boldsymbol{\nabla} \cdot \left(\rho(\boldsymbol{x}, t) \boldsymbol{\nabla} \boldsymbol{x}\right). \tag{5.25}$$

Here, $(1 - \gamma \boldsymbol{\nabla}^2)$, is the smoothing operator, $\boldsymbol{x} = (x, y)$, $\boldsymbol{\xi} = (\xi, \eta)$ and $\rho(\boldsymbol{x}, t)$ is the monitor function.

These equations are supplemented by boundary conditions:

$$\begin{aligned} &x(\xi = 0, \eta, t) = 0, \;\; x(\xi = 1, \eta, t) = L_x, \;\; x_\eta(\xi, \eta = 0, t) = x_\eta(\xi, \eta = 1, t) = 0, \\ &y(\xi, \eta = 0, t) = -L_y/2, \;\; y(\xi, \eta = 1, t) = L_y/2, \;\; y_\xi(\xi = 0, \eta, t) = y_\xi(\xi = 1, \eta, t) = 0. \end{aligned} \tag{5.26}$$

The initial conditions are taken to be a uniform mesh as follows

$$x_{j,k} = x\left(\xi_j, \eta_k, t\right) = L_x \xi_j, \;\; y_{j,k} = y\left(\xi_j, \eta_k, t\right) = L_y \eta_k - L_y/2. \tag{5.27}$$

The monitor function is chosen to be the curvature-based monitor function:

$$\rho(x, y, t) = \left(1 + \alpha |\boldsymbol{\nabla}^2 h|^2\right)^{\frac{1}{4}}, \tag{5.28}$$

where α is a user-specified parameter.

Finite difference discretisation of Eq. (5.18), MMPDEs and monitor function

A *conservative* semi-discretisation scheme for the spatial derivatives in Eq. (5.18) on the uniform mesh $\mathcal{J}_h^c(t)$ using centred finite differences can be written as, keeping the time derivative continuous,

$$h_{t,j,k} - (h_x x_t + h_y y_t)_{j,k} + \frac{1}{3}\boldsymbol{\nabla}\cdot\left[Cah^3\boldsymbol{\nabla}\boldsymbol{\nabla}^2 h - D(\theta)\ h^3\boldsymbol{\nabla}h\right]_{j,k} + \frac{1}{3}(h^3)_{x,j,k} = 0. \quad (5.29)$$

We first discretise the spatial derivatives, so that

$$\begin{aligned}
\left[h_x^3\right]_{j,k} =& \frac{1}{J}\left[\frac{1}{\Delta\xi}\left((J\xi_x)_{j+\frac{1}{2},k}\left(h^3\right)_{j+\frac{1}{2},k} - (J\xi_x)_{j-\frac{1}{2},k}\left(h^3\right)_{j-\frac{1}{2},k}\right) + \right. \\
& \left.\frac{1}{\Delta\eta}\left((J\eta_x)_{j,k+\frac{1}{2}}\left(h^3\right)_{j,k+\frac{1}{2}} - (J\eta_x)_{j,k-\frac{1}{2}}\left(h^3\right)_{j,k-\frac{1}{2}}\right)\right], \quad (5.30) \\
\left[h_x\right]_{j,k} =& \frac{1}{J}\left[\frac{1}{\Delta\xi}\left((J\xi_x)_{j+\frac{1}{2},k}(h)_{j+\frac{1}{2},k} - (J\xi_x)_{j-\frac{1}{2},k}(h)_{j-\frac{1}{2},k}\right) + \right. \\
& \left.\frac{1}{\Delta\eta}\left((J\eta_x)_{j,k+\frac{1}{2}}(h)_{j,k+\frac{1}{2}} - (J\eta_x)_{j,k-\frac{1}{2}}(h)_{j,k-\frac{1}{2}}\right)\right], \quad (5.31) \\
\left[h_y\right]_{j,k} =& \frac{1}{J}\left[\frac{1}{\Delta\xi}\left((J\xi_y)_{j+\frac{1}{2},k}(h)_{j+\frac{1}{2},k} - (J\xi_y)_{j-\frac{1}{2},k}(h)_{j-\frac{1}{2},k}\right) + \right. \\
& \left.\frac{1}{\Delta\eta}\left((J\eta_y)_{j,k+\frac{1}{2}}(h)_{j,k+\frac{1}{2}} - (J\eta_y)_{j,k-\frac{1}{2}}(h)_{j,k-\frac{1}{2}}\right)\right], \quad (5.32)
\end{aligned}$$

The vertical component of gravity and surface tension terms are discretised as

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}h\right]_{j,k}=\frac{1}{J}\left[\frac{(h^3R)_{j,k}-(h^3R)_{j-1,k}}{\Delta\xi}+\frac{(h^3S)_{j,k}-(h^3S)_{j,k-1}}{\Delta\eta}\right], \tag{5.33}$$

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}G\right]_{j,k}=\frac{1}{J}\left[\frac{(h^3R_1)_{j,k}-(h^3R_1)_{j-1,k}}{\Delta\xi}+\frac{(h^3S_1)_{j,k}-(h^3S_1)_{j,k-1}}{\Delta\eta}\right], \tag{5.34}$$

$$G_{j,k}=(\boldsymbol{\nabla}^2h)_{j,k}=\frac{1}{J}\left[\frac{(R)_{j,k}-(R)_{j-1,k}}{\Delta\xi}+\frac{(S)_{j,k}-(S)_{j,k-1}}{\Delta\eta}\right], \tag{5.35}$$

$$R_{j,k}=\left[a_{j+\frac{1}{2},k}\frac{h_{j+1,k}-h_{j,k}}{\Delta\xi}+c_{j+\frac{1}{2},k}\frac{h_{j,k+1}-h_{j,k}}{\Delta\eta}\right], \tag{5.36}$$

$$S_{j,k}=\left[q_{j,k+\frac{1}{2}}\frac{h_{j,k+1}-h_{j,k}}{\Delta\eta}+c_{j,k+\frac{1}{2}}\frac{h_{j+1,k}-h_{j,k}}{\Delta\xi}\right], \tag{5.37}$$

$$(R_1)_{j,k}=\left[a_{j+\frac{1}{2},k}\frac{G_{j+1,k}-G_{j,k}}{\Delta\xi}+c_{j+\frac{1}{2},k}\frac{G_{j,k+1}-G_{j,k}}{\Delta\eta}\right], \tag{5.38}$$

$$(S_1)_{j,k}=\left[q_{j,k+\frac{1}{2}}\frac{G_{j,k+1}-G_{j,k}}{\Delta\eta}+c_{j,k+\frac{1}{2}}\frac{G_{j+1,k}-G_{j,k}}{\Delta\xi}\right]. \tag{5.39}$$

We evaluate a, q, c and the Jacobian J as follows

$$a_{j,k} := \begin{cases} \frac{1}{J_{j,k}} \left((J\xi_x)^2 + (J\xi_y)^2 \right)_{j,k}, & \text{for } j = 2, \ldots, M_x, \\ \frac{1}{J_{j,k}} \left(J\xi_x \right)^2_{j,k}, & \text{for } j = 1, \\ \frac{1}{J_{j,k}} \left(J\xi_x \right)^2_{j,k}, & \text{for } j = M_x + 1, \end{cases} \tag{5.40}$$

$$q_{j,k} := \begin{cases} \frac{1}{J_{j,k}} \left((J\eta_y)^2 + (J\eta_x)^2 \right)_{j,k}, & \text{for } k = 2, \ldots, N_y, \\ \frac{1}{J_{j,k}} \left(J\eta_y \right)^2_{j,k}, & \text{for } k = 1, \\ \frac{1}{J_{j,k}} \left(J\eta_y \right)^2_{j,k}, & \text{for } k = N_y + 1, \end{cases} \tag{5.41}$$

$$c_{j,k} = \frac{1}{J_{j,k}} \left(\left(J\eta_y \right) \left(J\xi_y \right) + \left(J\eta_x \right) \left(J\xi_x \right) \right)_{j,k}, \quad \text{for } j = 2, \ldots, M_x, \quad k = 2, \ldots, N_y, \tag{5.42}$$

$$J_{j,k} := \begin{cases} \left(J\xi_x \right)_{j,k} \left(J\eta_y \right)_{j,k} - \left(J\xi_y \right)_{j,k} \left(J\eta_x \right)_{j,k}, & \text{for } j = 2, \ldots, M_x, \quad k = 2, \ldots, N_y, \\ \left(J\xi_x \right)_{j,k} \left(J\eta_y \right)_{j,k}, & \text{for } j, k = 1 \\ \left(J\xi_x \right)_{j,k} \left(J\eta_y \right)_{j,k}, & \text{for } j = M_x + 1, \quad k = N_y + 1. \end{cases} \tag{5.43}$$

We use centred finite difference approximations to approximate $J\xi_x$, $J\xi_y$, $J\eta_x$ and $J\eta_y$ as follows

$$\left(J\xi_x \right)_{j,k} = + \left(y_\eta \right)_{j,k} = + \frac{1}{2\Delta\eta} \left(y_{j,k+1} - y_{j,k-1} \right), \tag{5.44}$$

$$\left(J\xi_y \right)_{j,k} = - \left(x_\eta \right)_{j,k} = - \frac{1}{2\Delta\eta} \left(x_{j,k+1} - x_{j,k-1} \right), \tag{5.45}$$

$$\left(J\eta_x \right)_{j,k} = - \left(y_\xi \right)_{j,k} = - \frac{1}{2\Delta\xi} \left(y_{j+1,k} - y_{j-1,k} \right), \tag{5.46}$$

$$\left(J\eta_y \right)_{j,k} = + \left(x_\xi \right)_{j,k} = + \frac{1}{2\Delta\xi} \left(x_{j+1,k} - x_{j-1,k} \right), \tag{5.47}$$

where the transformation relations Eq. (5.20) have been used. The approximations Eqs. (5.36, 5.39) to the vertical component of gravity and surface tension terms involve evaluation of a, q, and c at half points. These evaluations have been obtained as an average of the neighbouring mesh points.

We discretise the MMPDEs Eq. (5.24) and Eq. (5.25), respectively, for $j = 2, \ldots, M_x$ and $k = 2, \ldots, N_y$, as follows

Regularised MMPDE5:

$$\begin{aligned} \tau \Bigg[\boldsymbol{x}_t - \gamma_1 \Bigg(& \frac{1}{\Delta\xi^2} \left(\boldsymbol{x}_{t,j+1,k} - 2\boldsymbol{x}_{t,j,k} + \boldsymbol{x}_{t,j-1,k} \right) \\ & - \frac{1}{\Delta\eta^2} \left(\boldsymbol{x}_{t,j,k+1} - 2\boldsymbol{x}_{t,j,k} + \boldsymbol{x}_{t,j,k-1} \right) \Bigg) \Bigg] = E_{j,k}, \end{aligned} \tag{5.48}$$

MMPDE4:

$$\begin{aligned} & \tau \left[\tfrac{1}{\Delta\xi^2} \left(\rho_{j+\frac{1}{2},k} \left(\boldsymbol{x}_{t,j+1,k} - \boldsymbol{x}_{t,j,k} \right) - \rho_{j-\frac{1}{2},k} \left(\boldsymbol{x}_{t,j,k} - \boldsymbol{x}_{t,j-1,k} \right) \right) \right. \\ & \left. - \tfrac{1}{\Delta\eta^2} \left(\rho_{j,k+\frac{1}{2}} \left(\boldsymbol{x}_{t,j,k+1} - \boldsymbol{x}_{t,j,k} \right) - \rho_{j,k-\frac{1}{2}} \left(\boldsymbol{x}_{t,j,k} - \boldsymbol{x}_{t,j,k-1} \right) \right) \right] = -E_{j,k}, \end{aligned} \tag{5.49}$$

where

$$\begin{aligned} E_{j,k} = \; & \tau \left[\tfrac{1}{\Delta\xi^2} \left(\rho_{j+\frac{1}{2},k} \left(\boldsymbol{x}_{j+1,k} - \boldsymbol{x}_{j,k} \right) - \rho_{j-\frac{1}{2},k} \left(\boldsymbol{x}_{j,k} - \boldsymbol{x}_{j-1,k} \right) \right) \right. \\ & \left. - \tfrac{1}{\Delta\eta^2} \left(\rho_{j,k+\frac{1}{2}} \left(\boldsymbol{x}_{j,k+1} - \boldsymbol{x}_{j,k} \right) - \rho_{j,k-\frac{1}{2}} \left(\boldsymbol{x}_{j,k} - \boldsymbol{x}_{j,k-1} \right) \right) \right], \end{aligned} \tag{5.50}$$

and $\boldsymbol{x} = (x, y)$.

The curvature-based monitor function Eq. (5.28) is discretised as follows:

$$\rho_{j,k} = (1 + \alpha|(\boldsymbol{\nabla}^2 h)_{j,k}|^2)^{\frac{1}{4}}, \tag{5.51}$$

where $(\boldsymbol{\nabla}^2 h)_{j,k}$ is approximated by Eq. (5.35). Smoothing of the discretised monitor function is done based on a two-dimensional generalisation of Eq. (3.40).

The boundary conditions $h(0, y, t) = 1$ and $h(L_x, y, t) = b$ are replaced by their ODE form:

$$h_{t,1,k} = h_{t,M_x+1,k} = 0, \ \ \forall \ \ k = 1, \ldots, N_y + 1. \tag{5.52}$$

Similarly, the boundary conditions $x(0, \eta, t) = 0$, $x(1, \eta, t) = L_x$, $y(\xi, 0, t) = -L_y/2$ and $y(\xi, 1, t) = L_y/2$ are replaced by their ODE form:

$$x_{t,1,k} = x_{t,M_x+1,k} = 0, \ \forall \ k = 1, \ldots, N_y + 1; \ y_{t,j,1} = y_{t,j,N_y+1} = 0, \ \forall \ j = 1, \ldots, M_x + 1 \tag{5.53}$$

Evaluating the above at the boundaries require fictitious points which are obtained by discretising the boundary conditions. In some cases, we had to use one-sided finite differences to evaluate a particular quantity at the boundary which was obtained using a Taylor's series approximation there.

The above semi-discretisations for h and $\boldsymbol{x} = (x, y)$ form a coupled system of $3(M_x + 1)(N_y+1)$ ordinary differential equations with initial condition given by Eqs. (5.4,5.27). These are solved using DASPK [12]. We again follow a staggered system of numbering

the unknowns which reduces the bandwidth of the system. We note here that a non-uniform initial mesh had to be used instead of the initial mesh in Eq. (5.27) which had an influence on the solution and the performance of the solver in comparison to the uniform initial mesh. This nonuniform initial mesh was obtained by solving in pseudo-time the chosen MMPDE (with the uniform mesh as the initial condition) with h fixed (hence, the monitor function ρ is also fixed) at its initial condition given by Eq. (5.4). This mesh was then used to solve the MMPDEs in real time. For the Regularised MMPDE5 equation, we had to choose the parameter values $\tau = 1$ and $\gamma_1 = \sqrt{\max(\rho)}$ for the pseudo-time calculation after which $\tau = 10^{-2}$ was chosen for the solution in real time.

In all the results presented below, the parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_x = 20$ and $L_y = 2$, $\alpha = 1$. We only present results using MMPDE4 with $\tau = 10^{-2}$; the regularised MMPDE5 gave similar results but was very stiff. Figure 5.5(a, b) show the surface and contour plots, respectively, of the initial condition given by Eq. (5.4) with $k = n = 1$, $m = 2$, $a_1 = 0.1$, $a_2 = 0.01$, $K_1 = 20$, $K_2 = 100$, $b = 0.1$, $L_x = 20$ and $L_y = 2$. Figure 5.5(c) shows the corresponding adaptive mesh obtained by solving MMPDE4 with $\tau = 10^{-2}$ in pseudo-time using the curvature-based monitor function with $\alpha = 1$. We observe that the mesh adapts to the initial condition predominantly in the x direction with more points at the leading edge, the mesh in the y-direction is uniform.

Figure 5.6(a, b) illustrate the surface plots (side and top view, respectively) of $h(x, y, t)$

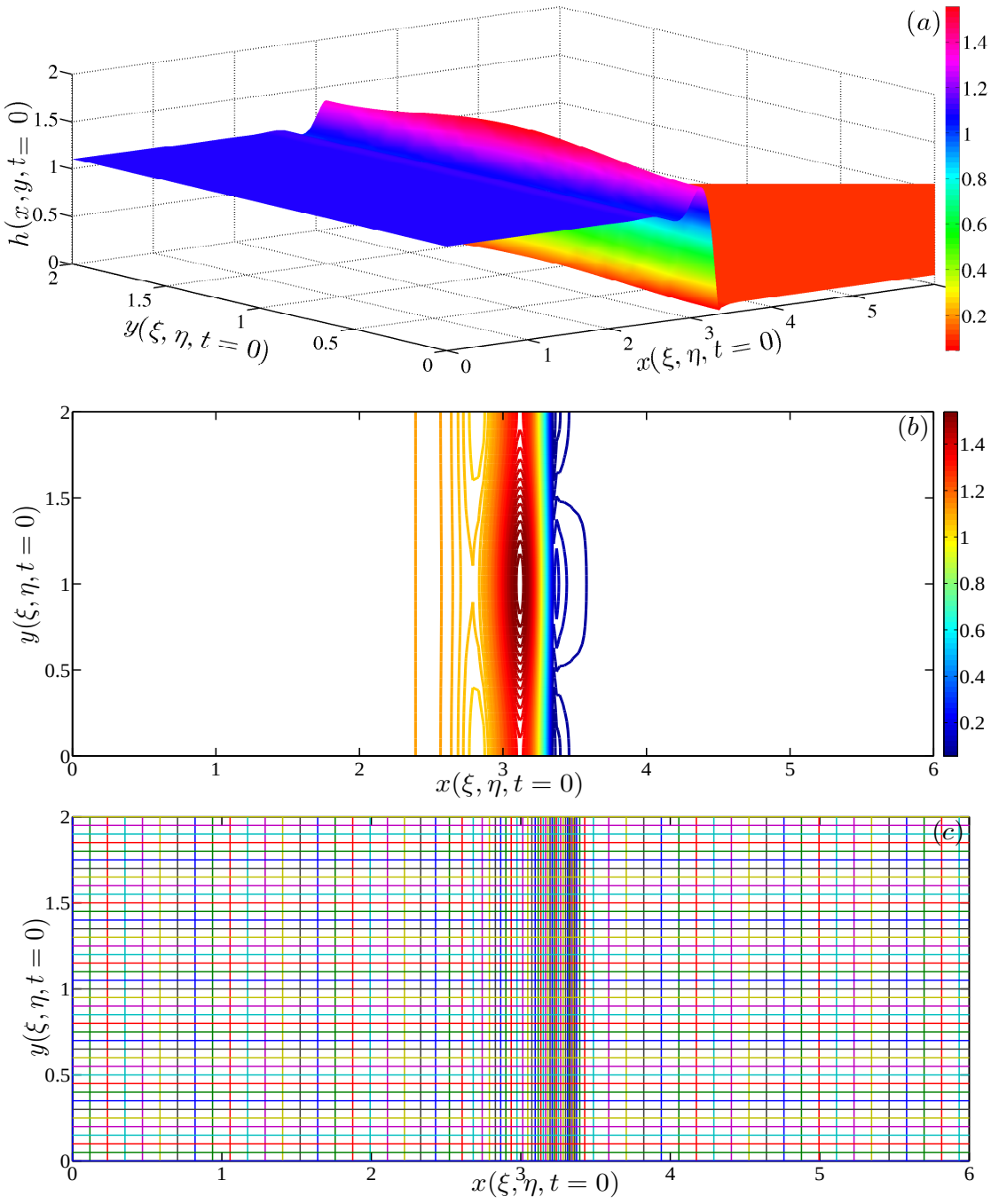

Figure 5.5: (a) Surface and (b) contour plots showing the two-dimensional initial condition given by Eq. (5.4) with $k = n = 1$, $m = 2$, $a_1 = 0.1$, $a_2 = 0.01$, $K_1 = 20$, $K_2 = 100$, $b = 0.1$, $L_x = 20$ and $L_y = 2$. (c) shows the corresponding adaptive mesh obtained by solving MMPDE4 with $\tau = 10^{-2}$ in pseudo-time (see text for details) using the curvature-based monitor function with $\alpha = 1$.

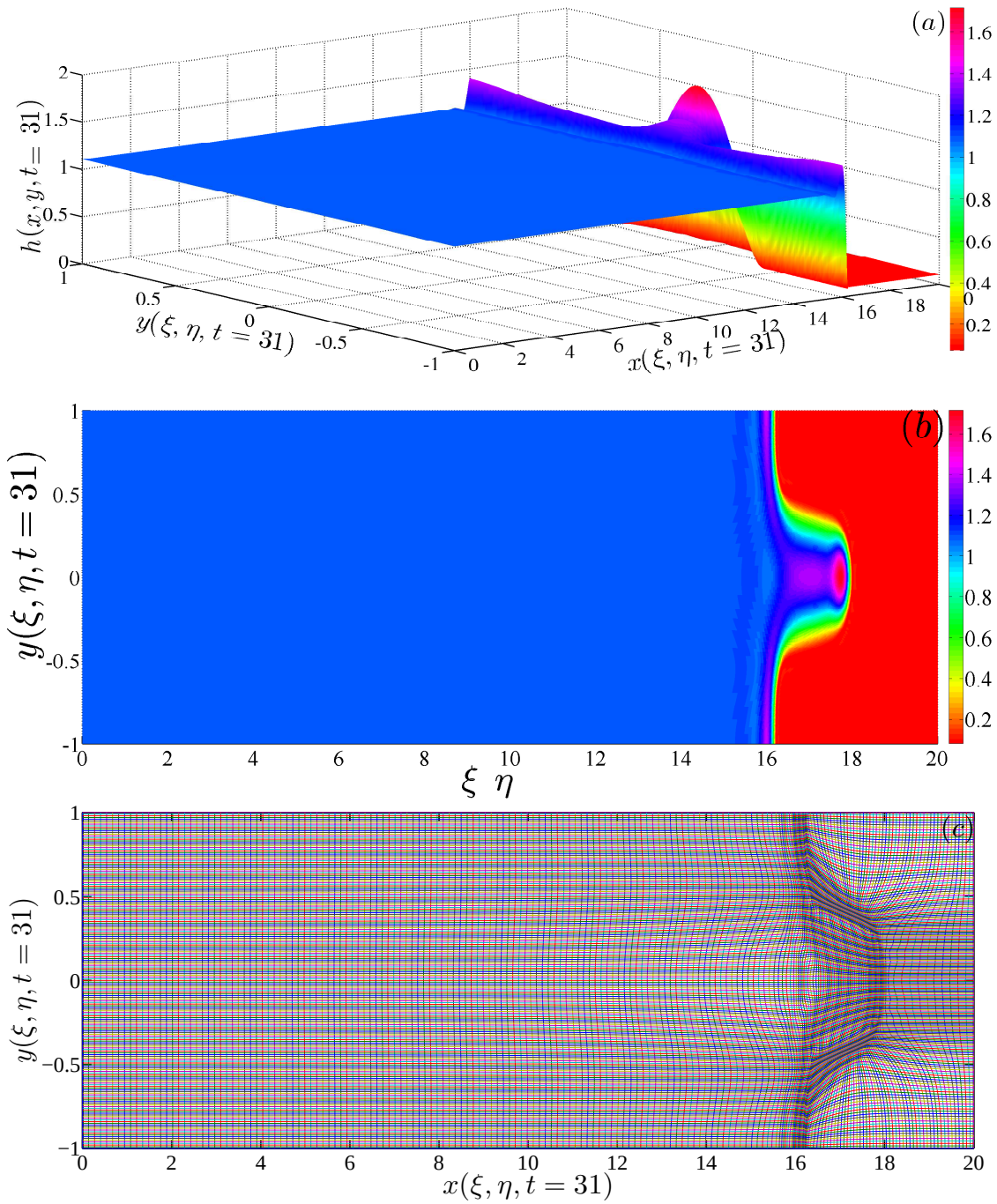

Figure 5.6: Surface plots ((*a*) side view and (*b*) top view) of $h(x, y, t)$ at $t = 31$ using the adaptive moving mesh scheme with $M_x = N_y = 200$ (so, initial $\Delta x = 0.1$ and $\Delta y = 10^{-2}$). (*c*) shows the corresponding adaptive moving mesh obtained using MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

at time $t = 31$ using the adaptive moving mesh scheme with $M_x = N_y = 200$ (so, initial $\Delta x = 0.1$ and $\Delta y = 10^{-2}$), MMPDE4 with $\tau = 10^{-2}$ and curvature monitor function with $\alpha = 1$. We clearly observe the formation of a single finger and visually it appears identical to the uniform mesh solution shown in figure 5.2(a, b) (note $M_x = 2000$ and $N_y = 200$ for this solution). Figure 5.6(c) shows the corresponding adaptive moving mesh and we clearly see clustering of mesh points in the y-direction along the finger.

This can also be observed from figure 5.7(a, b) which plots the adaptive mesh $y(\xi, \eta)$ and $x(\xi, \eta)$, respectively at $t = 31$. We see that y when considered as a function of η for fixed $\xi \in [0.6, 0.8]$, deviates from a straight line (representing a uniform mesh) with more points clustered in the interval $[-0.5, 0.5]$ which is approximately the width of the finger (see figure 5.6(b, c)). y appears to be uniform in the ξ direction. Similarly, x varies nonuniformly as a function of ξ with more points clustered around $\xi \in [0.6, 0.8]$ which is is approximately the finger lengthscale. x appears to be uniform in the η direction. Figure 5.7(c) shows the curvature-based monitor function at $t = 31$ which has large variations in the curvature near the finger region. This results in the refinement of the mesh as seen in figure 5.7(a, b).

Figure 5.8(a, b, c) illustrate the surface plots of $h(x, y, t)$ showing its evolution in time (times shown are $t = 11$, $t = 20$ and $t = 31$). At early time $t = 11$, a finger is observed to slowly start forming. As time t increases, the finger appears to develop with a preferred width. This is more clearly depicted in the contour plots presented in figure 5.9(a, b, c) where the width of the finger is approximately one and it lengthens as time

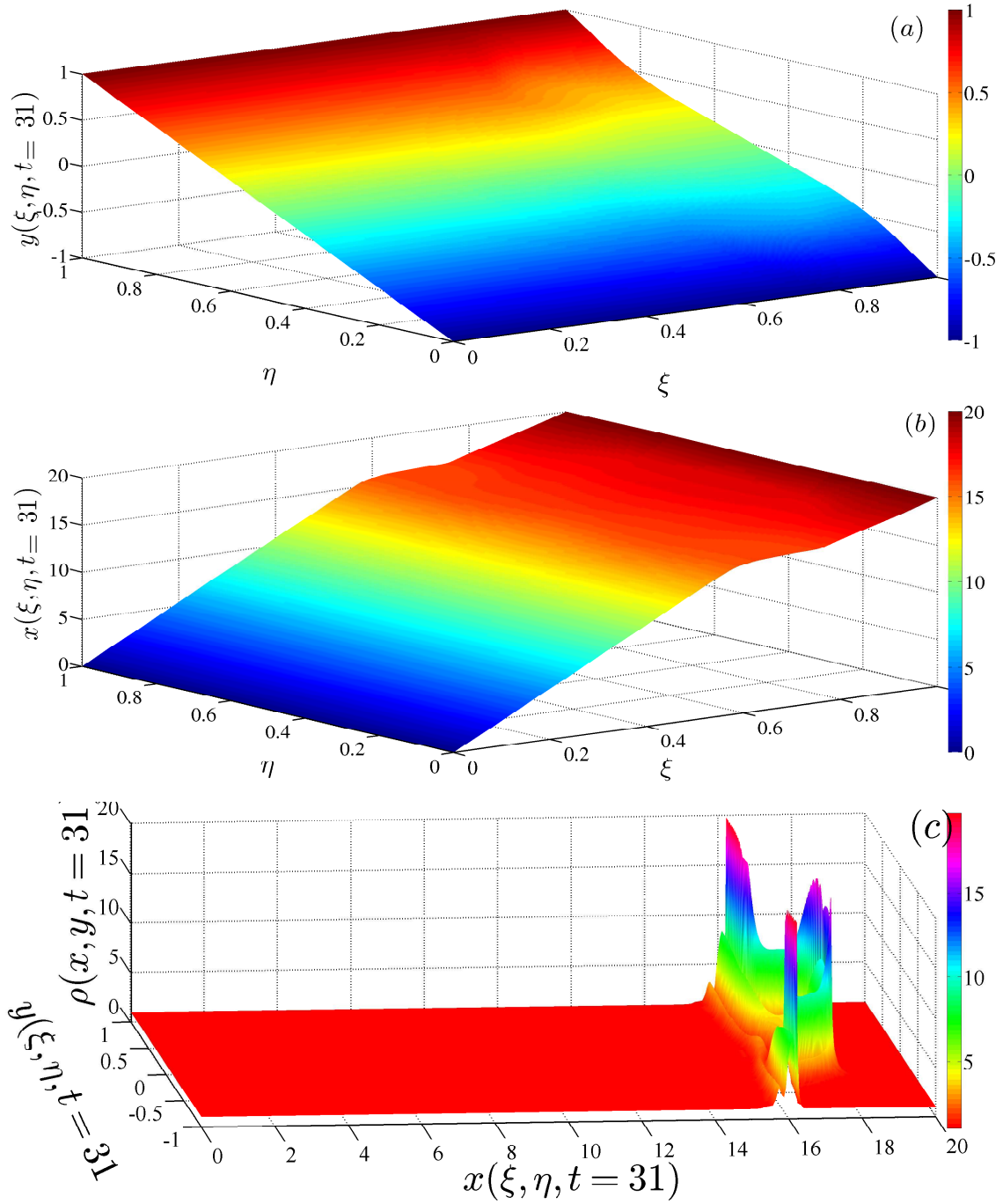

Figure 5.7: Surface plots showing the adaptive mesh (a) $y(\xi,\eta)$ and (b) $x(\xi,\eta)$, and the monitor function $\rho(x,y,t)$ corresponding to figure 5.6 at $t=31$ using the adaptive moving mesh scheme with $M_x = N_y = 200$ (so, initial $\Delta x = 0.1$ and $\Delta y = 10^{-2}$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

t increases. These results are indistinguishable from those obtained using a uniform mesh (see figures 5.3, 5.4). Figure 5.10(a, b, c) show the adaptive moving mesh at times $t = 11$, $t = 20$ and $t = 31$, respectively. We clearly see the mesh adaptation in both x and y as the propagating finger gradually develops.

5.1.3 Numerical solution of Eqs. (5.1,5.2,5.3) using the Parabolic-Mongè-Ampere (PMA) equation

Consider the transformed PDE Eq. (5.18) with the boundary conditions Eqs. (5.2,5.3) and the initial condition Eq. (5.4). Expressing the mesh $\boldsymbol{x}$ and the velocity $\boldsymbol{x}_t$ in Eq. (5.18) in terms of Q, leads to

$$h_t - (h_x Q_{t,\xi} + h_y Q_{t,\eta}) + \boldsymbol{\nabla} \cdot [Ca \frac{h^3}{3} \boldsymbol{\nabla} \boldsymbol{\nabla}^2 h - D(\theta) \frac{h^3}{3} \boldsymbol{\nabla} h] + \left(\frac{h^3}{3} \right)_x = 0. \tag{5.54}$$

Substituting the transformation relations Eq. (5.20) into Eq. (5.21), we have

$$\begin{bmatrix} h_x \\ h_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} y_\eta & -y_\xi \\ -x_\eta & x_\xi \end{bmatrix} \begin{bmatrix} h_\xi \\ h_\eta \end{bmatrix}. \tag{5.55}$$

Thus, the transformations are obtained using the fact that $\boldsymbol{x} = \boldsymbol{\nabla}_{\boldsymbol{\xi}} Q$ by

$$h_x = \frac{1}{H} \left[Q_{\eta\eta} h_\xi - Q_{\xi\eta} h_\eta \right], \tag{5.56}$$

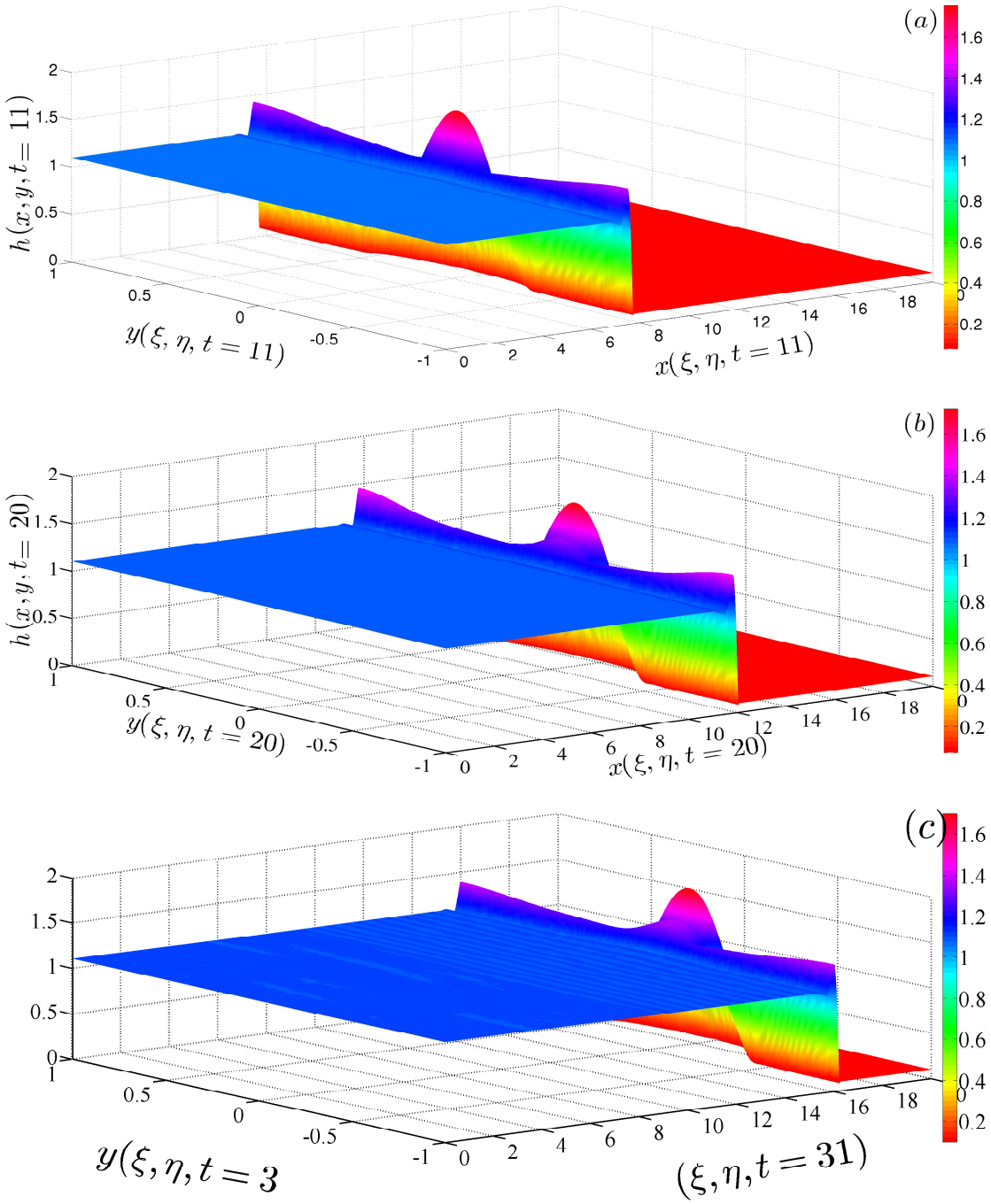

Figure 5.8: Surface plots of $h(x, y, t)$ showing its evolution in time (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$ using the adaptive moving mesh scheme with $M_x = N_y = 200$ (so, initial $\Delta x = 0.1$ and $\Delta y = 10^{-2}$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

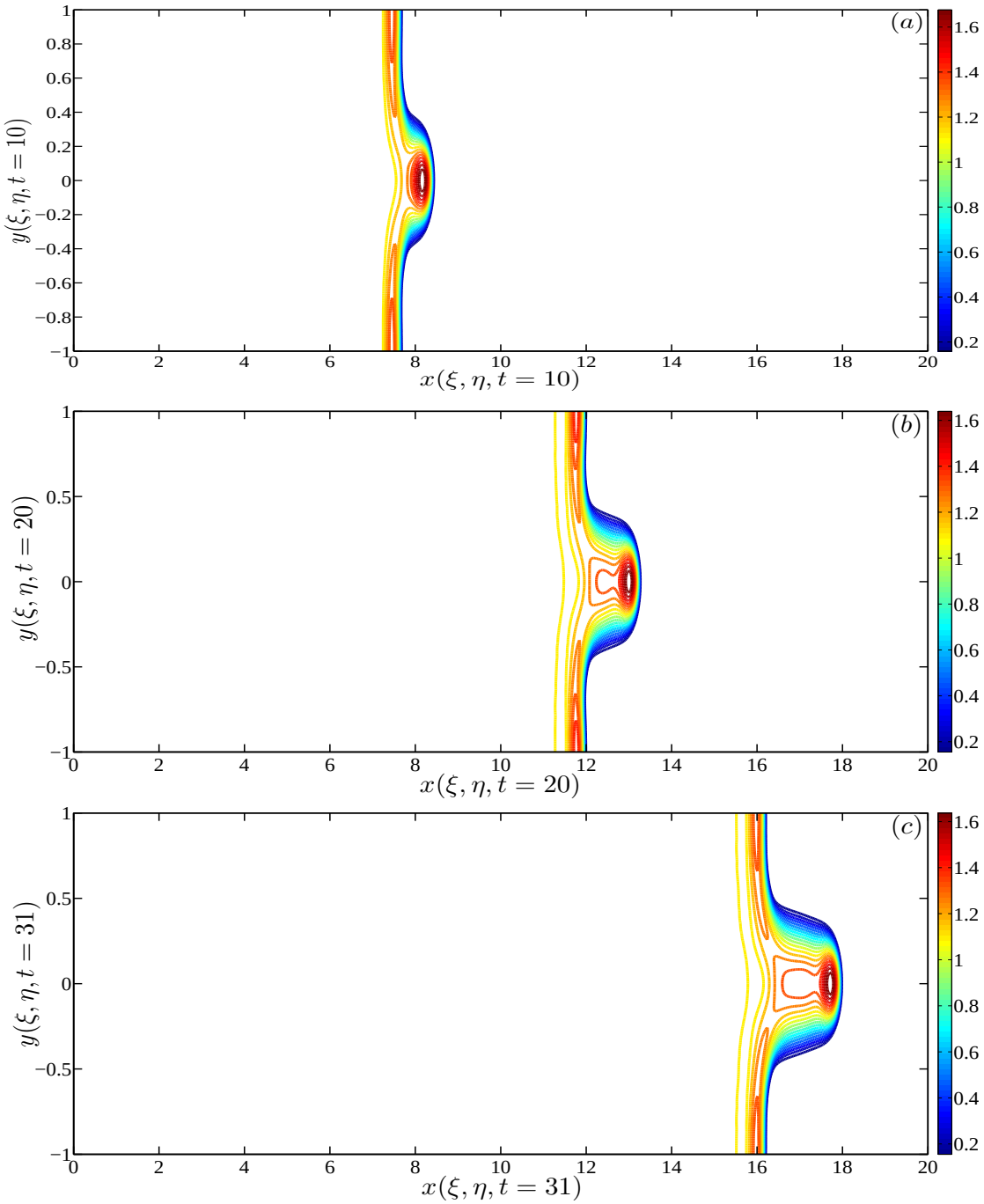

Figure 5.9: Contour plots of $h(x,y,t)$ showing its evolution in time (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$ using the adaptive moving mesh scheme with $M_x = N_y = 200$ (so, initial $\Delta x = 0.1$ and $\Delta y = 10^{-2}$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

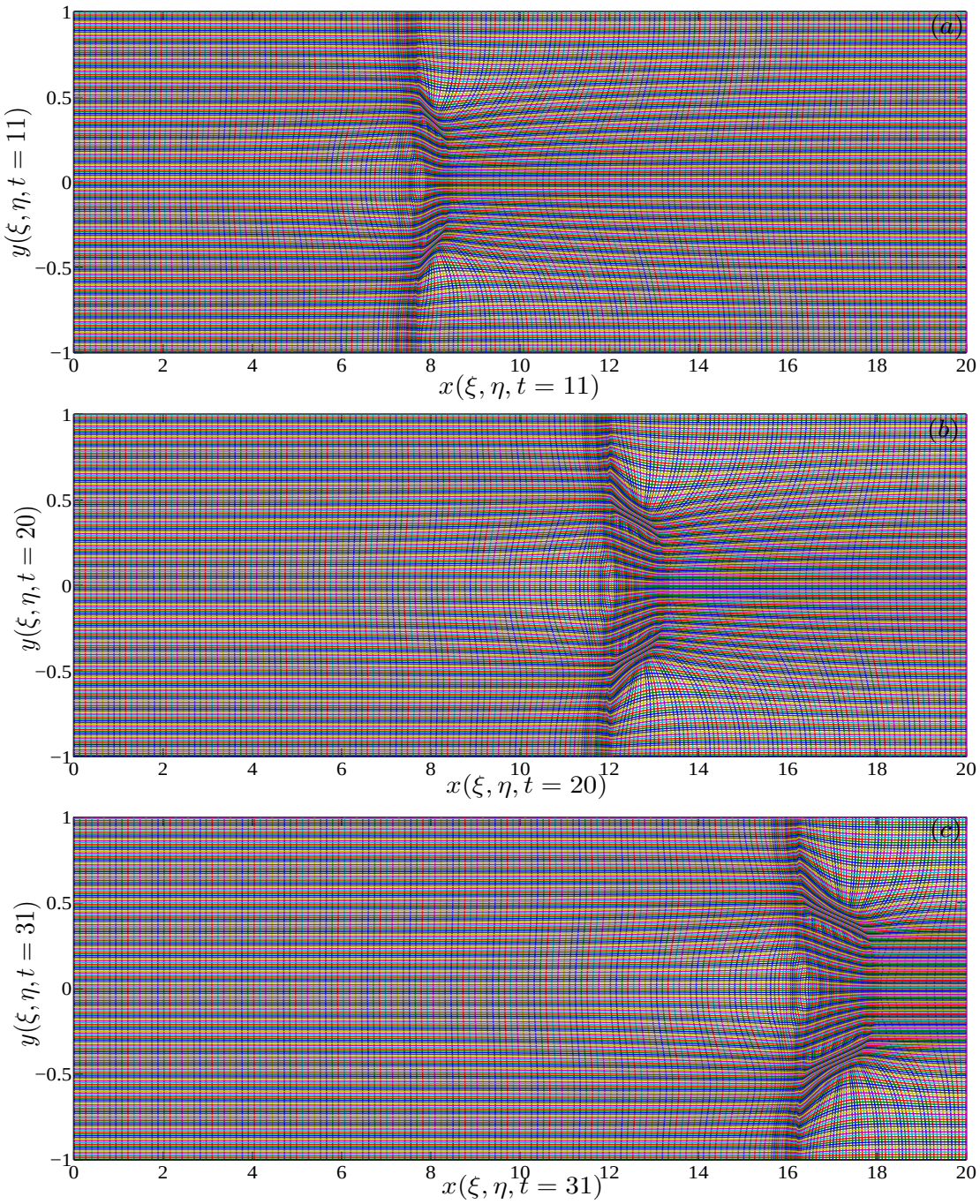

Figure 5.10: The adaptive moving mesh showing its evolution in time (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$ using the adaptive moving mesh scheme with $M_x = N_y = 200$ (so, initial $\Delta x = 0.1$ and $\Delta y = 10^{-2}$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

$$h_y = \frac{1}{H}\left[-Q_{\xi\eta}h_\xi + Q_{\xi\xi}h_\eta\right], \tag{5.57}$$

where $\boldsymbol{\nabla_\xi} = \left(\frac{\partial}{\partial\xi}, \frac{\partial}{\partial\eta}\right)^T$ and $\boldsymbol{H}$ is the Hessian of Q, which is equivalent to the Jacobian $\boldsymbol{J}$. The Hessian is given by:

$$\boldsymbol{H}(Q) = \begin{bmatrix} Q_{\xi\xi} & Q_{\xi\eta} \\ Q_{\xi\eta} & Q_{\eta\eta} \end{bmatrix}, \quad H \equiv \det(\boldsymbol{H}) = Q_{\xi\xi}Q_{\eta\eta} - Q_{\xi\eta}^2. \tag{5.58}$$

The spatial derivatives of Eq. (5.54) are transformed as follows:

$$h_x^3 = \frac{1}{H}\left[Q_{\eta\eta}h_\xi^3 - Q_{\xi\eta}h_\eta^3\right]. \tag{5.59}$$

Since $h_{xx} = (h_x)_x$ and $h_{yy} = (h_y)_y$, the vertical component of gravity and the surface tension terms can be written as

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}h\right] = \frac{1}{H}\left[\left(h^3R\right)_\xi + \left(h^3S\right)_\eta\right], \tag{5.60}$$

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}G\right] = \frac{1}{H}\left[\left(h^3R_1\right)_\xi + \left(h^3S_1\right)_\eta\right], \tag{5.61}$$

$$G = \boldsymbol{\nabla}^2 h = \frac{1}{H}\left[(R)_\xi + (S)_\eta\right], \tag{5.62}$$

$$R = \left[ah_\xi - ch_\eta\right],\ S = \left[qh_\eta - ch_\xi\right],\ R_1 = \left[aG_\xi - cG_\eta\right],\ S_1 = \left[qG_\eta - cG_\xi\right], \tag{5.63}$$

$$a = \frac{1}{H}\left(Q_{\eta\eta}^2 + Q_{\xi\eta}^2\right),\ \ q = \frac{1}{H}\left(Q_{\xi\xi}^2 + Q_{\xi\eta}^2\right),\ \ c = \frac{1}{H}\left(Q_{\xi\eta}Q_{\eta\eta} + Q_{\xi\eta}Q_{\xi\xi}\right). \tag{5.64}$$

Parabolic-Mongè-Ampere (PMA) equation in two dimensions

We use the two-dimensional PMA equation described in Chapter 2. This is given by

$$\tau \left(1 - \gamma_1 \boldsymbol{\nabla}_{\boldsymbol{\xi}}^2\right) Q_t = \left(\rho(\boldsymbol{\nabla}_{\boldsymbol{\xi}} Q, t) H(Q)\right)^{\frac{1}{2}}, \quad \boldsymbol{x} = \boldsymbol{\nabla}_{\boldsymbol{\xi}} Q. \tag{5.65}$$

where $\boldsymbol{x} = (x, y)$ and $\boldsymbol{\xi} = (\xi, \eta)$.

This equation is supplemented by Neumann boundary conditions, which are

$$[Q_\xi]_{j,k} = 0, L_x, \quad \text{for} \quad j = 1, M_x + 1, \quad [Q_\eta]_{j,k} = -L_y/2, L_y/2, \quad \text{for} \quad k = 1, N_y + 1. \tag{5.66}$$

The initial condition is taken to be a uniform mesh as follows:

$$Q(\xi, \eta, 0) = \frac{1}{2} \left(L_x \xi^2 + L_y \eta^2\right). \tag{5.67}$$

We choose the monitor function to be the curvature-based monitor function:

$$\rho \left(1 + \alpha |\boldsymbol{\nabla}^2 h|^2\right)^{\frac{1}{4}}, \tag{5.68}$$

where α is a user-specified parameters.

Finite difference discretisation of Eq. (5.54) and PMA equation

A *conservative* semi-discretisation scheme for the spatial derivatives in Eq. (5.54) on the uniform mesh $\mathcal{J}_h^c(t)$ using centred finite differences can be written as, keeping the time derivative continuous,

$$h_{t,j,k} - [h_x]_{j,k} \frac{Q_{t,j+1,k} - Q_{t,j-1,k}}{2\Delta\xi} - [h_y]_{j,k} \frac{Q_{t,j,k+1} - Q_{t,j,k-1}}{2\Delta\eta} + \frac{1}{3}\boldsymbol{\nabla} \cdot \left[Cah^3 \boldsymbol{\nabla}\boldsymbol{\nabla}^2 h - D(\theta)h^3 \boldsymbol{\nabla} h\right]_{j,k} + \frac{1}{3}(h^3)_{x,j,k} = 0. \tag{5.69}$$

The first spatial derivatives are discretised by

$$\left[h_x^3\right]_{j,k} = \frac{1}{H}\left[\frac{1}{\Delta\xi}\left((Q_{\eta\eta})_{j+\frac{1}{2},k}\left(h^3\right)_{j+\frac{1}{2},k} - (Q_{\eta\eta})_{j-\frac{1}{2},k}\left(h^3\right)_{j-\frac{1}{2},k}\right) + \frac{1}{\Delta\eta}\left(-(Q_{\eta\xi})_{j,k+\frac{1}{2}}\left(h^3\right)_{j,k+\frac{1}{2}} + (Q_{\eta\xi})_{j,k-\frac{1}{2}}\left(h^3\right)_{j,k-\frac{1}{2}}\right)\right], \tag{5.70}$$

$$[h_x]_{j,k} = \frac{1}{H}\left[\frac{1}{\Delta\xi}\left((Q_{\eta\eta})_{j+\frac{1}{2},k}(h)_{j+\frac{1}{2},k} - (Q_{\eta\eta})_{j-\frac{1}{2},k}(h)_{j-\frac{1}{2},k}\right) + \tag{5.71}$$

$$\frac{1}{\Delta\eta}\left(-(Q_{\eta\xi})_{j,k+\frac{1}{2}}(h)_{j,k+\frac{1}{2}} + (Q_{\eta\xi})_{j,k-\frac{1}{2}}(h)_{j,k-\frac{1}{2}}\right)\right], \tag{5.72}$$

$$[h_y]_{j,k} = \frac{1}{H}\left[\frac{1}{\Delta\xi}\left(-(Q_{\eta\xi})_{j+\frac{1}{2},k}(h)_{j+\frac{1}{2},k} + (Q_{\eta\xi})_{j-\frac{1}{2},k}(h)_{j-\frac{1}{2},k}\right) + \tag{5.73}$$

$$\frac{1}{\Delta\eta}\left((Q_{\xi\xi})_{j,k+\frac{1}{2}}(h)_{j,k+\frac{1}{2}} - (Q_{\xi\xi})_{j,k-\frac{1}{2}}(h)_{j,k-\frac{1}{2}}\right)\right], \tag{5.74}$$

The vertical component of gravity and surface tension terms are discretised by

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}h\right] = \frac{1}{H_{j,k}}\left[\frac{(h^3R)_{j,k}-(h^3R)_{j-1,k}}{\Delta\xi}+\frac{(h^3S)_{j,k}-(h^3S)_{j,k-1}}{\Delta\eta}\right], \tag{5.75}$$

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}G\right] = \frac{1}{H_{j,k}}\left[\frac{(h^3R_1)_{j,k}-(h^3R_1)_{j-1,k}}{\Delta\xi}+\frac{(h^3S_1)_{j,k}-(h^3S_1)_{j,k-1}}{\Delta\eta}\right], \tag{5.76}$$

$$G = \boldsymbol{\nabla}^2 h = \frac{1}{H_{j,k}}\left[\frac{(R)_{j,k}-(R)_{j-1,k}}{\Delta\xi}+\frac{(S)_{j,k}-(S)_{j,k-1}}{\Delta\eta}\right], \tag{5.77}$$

$$R_{j,k} = \left[a_{j+\frac{1}{2},k}\frac{h_{j+1,k}-h_{j,k}}{\Delta\xi}-c_{j+\frac{1}{2},k}\frac{h_{j,k+1}-h_{j,k}}{\Delta\eta}\right], \tag{5.78}$$

$$S_{j,k} = \left[q_{j,k+\frac{1}{2}}\frac{h_{j,k+1}-h_{j,k}}{\Delta\eta}-c_{j,k+\frac{1}{2}}\frac{h_{j+1,k}-h_{j,k}}{\Delta\xi}\right], \tag{5.79}$$

$$(R_1)_{j,k} = \left[a_{j+\frac{1}{2},k}\frac{G_{j+1,k}-G_{j,k}}{\Delta\xi}-c_{j+\frac{1}{2},k}\frac{G_{j,k+1}-G_{j,k}}{\Delta\eta}\right], \tag{5.80}$$

$$(S_1)_{j,k} = \left[q_{j,k+\frac{1}{2}}\frac{G_{j,k+1}-G_{j,k}}{\Delta\eta}-c_{j,k+\frac{1}{2}}\frac{G_{j+1,k}-G_{j,k}}{\Delta\xi}\right]. \tag{5.81}$$

We evaluate a, q, c and H as follows:

$$a_{j,k} := \begin{cases} \frac{1}{H_{j,k}} \left((Q_{\eta\eta})^2 + (Q_{\eta\xi})^2 \right)_{j,k}, & \text{for } j = 2, \ldots, M_x, \\ \frac{1}{H_{j,k}} (Q_{\eta\eta})^2_{j,k}, & \text{for } j = 1, \\ \frac{1}{H_{j,k}} (Q_{\eta\eta})^2_{j,k}, & \text{for } j = M_x + 1, \end{cases} \tag{5.82}$$

$$q_{j,k} := \begin{cases} \frac{1}{H_{j,k}} \left((Q_{\xi\xi})^2 + (Q_{\eta\xi})^2 \right)_{j,k}, & \text{for } k = 2, \ldots, N_y, \\ \frac{1}{H_{j,k}} (Q_{\xi\xi})^2_{j,k}, & \text{for } k = 1, \\ \frac{1}{H_{j,k}} (Q_{\xi\xi})^2_{j,k}, & \text{for } k = N_y + 1, \end{cases} \tag{5.83}$$

$$c_{j,k} = \frac{1}{H_{j,k}} Q_{\eta\xi} \left(Q_{\xi\xi} + Q_{\eta\eta} \right)_{j,k}, \text{ for } \quad j = 2, \ldots, M_x, \quad k = 2, \ldots, N_y, \tag{5.84}$$

$$H_{j,k} := \begin{cases} (Q_{\xi\xi})_{j,k} (Q_{\eta\eta})_{j,k} - (Q_{\eta\xi})^2_{j,k}, & \text{for } \quad j = 2, \ldots, M_x, \quad k = 2, \ldots, N_y, \\ (Q_{\xi\xi})_{j,k} (Q_{\eta\eta})_{j,k}, & \text{for } \quad j, k = 1, \\ (Q_{\xi\xi})_{j,k} (Q_{\eta\eta})_{j,k}, & \text{for } \quad j = M_x + 1, \quad k = N_y + 1, \cdot \end{cases} \tag{5.85}$$

The approximations Eqs. (5.78, 5.81) to the vertical component of gravity and surface tension terms involve evaluations for a, q, and c at half points. These evaluations have been obtained as an average of the neighbouring mesh point values. At the boundaries in x direction, the vertical component of gravity and surface tension terms require fictitious points which are obtained by discretising the boundary condtions $h_\xi = 0$ and $h_{\xi\xi\xi} = 0$ at $\xi = 0, 1$ using centred finite differences. While in the y direction, the boundary conditions $h_\eta = 0$ and $h_{\eta\eta\eta} = 0$ are discretised to obtain the fictitious points.

We obtain

$$h_{j-1,k} = h_{j+1,k},\ h_{j-2,k} = h_{j+2,k},\ \text{at } \xi = 0, 1,$$

$$h_{j,k-1} = h_{j,k+1},\ h_{j,k-2} = h_{j,k+2},\ \text{at } \eta = 0, 1.$$

The second order derivatives of Q are obtained using the centred finite differences at the interior points by

$$[Q_{\xi\xi}]_{j,k} = \frac{1}{2\Delta\xi^2}\left[Q_{j+2,k} - 2Q_{j,k} + Q_{j-2,k}\right], \quad \text{for } \ j = 3, \ldots, M_x - 1, \tag{5.86}$$

$$[Q_{\eta\eta}]_{j,k} = \frac{1}{2\Delta\eta^2}\left[Q_{j,k+2} - 2Q_{j,k} + Q_{j,k-2}\right], \quad \text{for } \ k = 3, \ldots, N_y - 1, \tag{5.87}$$

$$[Q_{\eta\xi}]_{j,k} = \frac{1}{4\Delta\eta\Delta\xi}\left[Q_{j+1,k+1} - Q_{j+1,k-1} - Q_{j-1,k+1} + Q_{j-1,k-1}\right], \tag{5.88}$$

$$\text{for } \ j = 2, \ldots, M_x, \ \ k = 2, \ldots, N_y.$$

To approximate $Q_{\xi\xi}$ and $Q_{\eta\eta}$ at $j = 2, M_x$ and $k = 2, N_y$, respectively, we use the fact that $Q_\xi = x$ and $Q_\eta = y$. We obtain

$$[Q_{\xi\xi}]_{2,k} = \frac{1}{4\Delta\xi^2}\left[Q_{4,k} - Q_{2,k}\right], \tag{5.89}$$

$$[Q_{\xi\xi}]_{M_x,k} = \frac{1}{4\Delta\xi^2}\left[2\ L_x\ \Delta\xi - Q_{M_x,k} + Q_{M_x-2,k}\right], \tag{5.90}$$

$$[Q_{\eta\eta}]_{j,2} = \frac{1}{4\Delta\eta^2}\left[Q_{j,4} - Q_{j,2} + L_y\Delta\eta\right], \tag{5.91}$$

$$[Q_{\eta\eta}]_{j,N_y} = \frac{1}{4\Delta\eta^2}\left[L_y\ \Delta\eta - Q_{j,N_y} + Q_{j,N_y-2}\right], \tag{5.92}$$

At the boundaries, we force $Q_{\eta\xi} = 0$. To evaluate $Q_{\xi\xi}$ at $\xi = 0$, we utilize the Taylor expansions so that

$$x_{2,k} = x_{1,k} + \Delta\xi \left[x_\xi\right]_{1,k} + \frac{1}{2}\Delta\xi^2 \left[x_{\xi\xi}\right]_{1,k} + (\Delta\xi^3), \tag{5.93}$$

$$x_{3,k} = x_{1,k} + 2\Delta\xi \left[x_\xi\right]_{1,k} + \frac{4}{2}\Delta\xi^2 \left[x_{\xi\xi}\right]_{1,k} + (\Delta\xi^3), \tag{5.94}$$

after some processing and neglecting terms of $O(\Delta\xi^3)$, we obtain

$$\left[x_\xi\right]_{1,k} = \frac{1}{2\Delta\xi}\left(4x_{2,k} - x_{3,k}\right). \tag{5.95}$$

Since $Q_\xi = x$, leads to

$$\left[Q_{\xi\xi}\right]_{1,k} = \frac{1}{4\Delta\xi^2}\left[4Q_{3,k} - 4Q_{1,k} - Q_{4,k} + Q_{2,k}\right], \quad \forall \ k. \tag{5.96}$$

Similarly, we approximate $Q_{\xi\xi}$ at $\xi = 1$ by

$$\left[Q_{\xi\xi}\right]_{M_x+1,k} = \frac{1}{4\Delta\xi^2}\left[Q_{M_x,k} - Q_{M_x-2,k} - 4Q_{M_x+1,k} + 4Q_{M_x-1,k} + 6\ L_x\ \Delta\xi\right], \quad \forall \ k. \tag{5.97}$$

We evaluate $Q_{\eta\eta}$ at $\eta = 0, 1$, in a similar manner so that

$$\left[Q_{\eta\eta}\right]_{j,1} = \frac{1}{4\Delta\eta^2}\left[4Q_{j,3} - 4Q_{j,1} - Q_{j,4} + Q_{j,2} + 3\ L_y\ \Delta\eta\right], \quad \forall \ j, \tag{5.98}$$

$$\left[Q_{\eta\eta}\right]_{j,N_y+1} = \frac{1}{4\Delta\eta^2}\left[Q_{j,N_y} - Q_{j,N_y-2} - 4Q_{j,N_y+1} + 4Q_{j,N_y-1} + 3\ L_y\ \Delta\eta\right], \quad \forall \ j. \tag{5.99}$$

We discretise the PMA equation Eq. (5.65) at the interior points of the computational domain using centred finite differences, so that

$$[Q_{\xi\xi}]_{j,k} = \frac{1}{\Delta\xi^2}\left[Q_{j+1,k} - 2Q_{j,k} + Q_{j-1,k}\right], \quad \text{for} \;\; j = 2,\ldots,M_x, \tag{5.100}$$

$$[Q_{\eta\eta}]_{j,k} = \frac{1}{\Delta\eta^2}\left[Q_{j,k+1} - 2Q_{j,k} + Q_{j,k-1}\right], \quad \text{for} \;\; k = 2,\ldots,N_y, \tag{5.101}$$

$$[Q_{\eta\xi}]_{j,k} = \frac{1}{4\Delta\eta\Delta\xi}\left[Q_{j+1,k+1} - Q_{j+1,k-1} - Q_{j-1,k+1} + Q_{j-1,k-1}\right], \tag{5.102}$$

$$\text{for} \;\; j = 2,\ldots,M_x, \;\; k = 2,\ldots,N_y.$$

The Hessian

$$H = Q_{\xi\xi}Q_{\eta\eta} - Q_{\eta\xi}^2, \tag{5.103}$$

is determined by using the above approximations of the second order derivatives of Q. At the boundaries, we force $Q_{\eta\xi} = 0$. We approximate $Q_{\xi\xi}$ at $x = 0$ using the following Taylor expansions of Q, so that

$$Q_{2,k} = Q_{1,k} + \Delta\xi\left[Q_{\xi}\right]_{1,k} + \frac{1}{2}\Delta\xi^2\left[Q_{\xi\xi}\right]_{1,k} + \frac{1}{6}\Delta\xi^3\left[Q_{\xi\xi}\right]_{1,k} + (\Delta\xi^4), \tag{5.104}$$

$$Q_{3,k} = Q_{1,k} + 2\Delta\xi\left[Q_{\xi}\right]_{1,k} + \frac{4}{2}\Delta\xi^2\left[Q_{\xi\xi}\right]_{1,k} + \frac{8}{6}\Delta\xi^3\left[Q_{\xi\xi}\right]_{1,k} + (\Delta\xi^4). \tag{5.105}$$

Since $[Q_{\xi}]_{1,k} = 0$ at $x = 0$, and after some processing with neglecting terms of $O(\Delta\xi^4)$, we obtain

$$[Q_{\xi\xi}]_{1,k} = \frac{1}{2\Delta\xi^2}\left[-7Q_{1,k} + 8Q_{2,k} - Q_{3,k}\right], \quad \forall \;\; k. \tag{5.106}$$

We evaluate $Q_{\xi\xi}$ at $x = L_x$, and $Q_{\eta\eta}$ at $y = \pm L_y/2$, in similar manner, this leads to

$$[Q_{\xi\xi}]_{M_x+1,k} = \frac{1}{2\Delta\xi^2}\left[-7Q_{M_x+1,k} + 8Q_{M_x,k} - Q_{M_x-1,k} + 6\ \Delta\xi\ L_x\right],\ \ \forall\ \ k, \tag{5.107}$$

$$[Q_{\eta\eta}]_{j,1} = \frac{1}{2\Delta\eta^2}\left[-7Q_{j,1} + 8Q_{j,2} - Q_{j,3} + 3\ L_y\ \Delta\eta\right],\ \ \forall\ \ j, \tag{5.108}$$

$$[Q_{\eta\eta}]_{j,N_y+1} = \frac{1}{2\Delta\eta^2}\left[-7Q_{j,N_y+1} + 8Q_{j,N_y} - Q_{j,N_y-1} + 3\ L_y\ \Delta\eta\right],\qquad \forall j. \tag{5.109}$$

The discretised curvature-based monitor function Eq. (5.51) is used. The above semi-discretisations for h and Q form a coupled system of $2(M_x + 1)(N_y + 1)$ ordinary differential equations with initial condition given by Eqs. (5.4,5.67). These are solved using DASPK [12]. We again follow a staggered system of numbering the unknowns which reduces the bandwidth of the system. We note here that a non-uniform initial mesh had to be used instead of the initial mesh in Eq. (5.67) which had an influence on the solution and the performance of the solver in comparison to the uniform initial mesh. This nonuniform initial mesh was obtained by solving in pseudo-time the PMA equation (with the uniform mesh as the initial condition) with h fixed (hence, the monitor function ρ is also fixed) at its initial condition given by Eq. (5.4). This mesh was then used to solve the MMPDEs in real time.

Figure 5.11(a, b, c) illustrate the surface plots of $h(x, y, t)$ showing its evolution in time (times shown are $t = 11$, $t = 20$ and $t = 31$). These numerical solutions are obtained using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$. At

early time $t = 11$, a finger is observed to slowly start forming. As time t increases, the finger appears to develop with a preferred width. This is more clearly depicted in the contour plots presented in figure $5.12(a, b, c)$ where the width of the finger is approximately one and it lengthens as time t increases. These results are similar to those obtained using both a uniform mesh (see figures 5.3, 5.4) and adaptive moving mesh using MMPDE4 (see figures 5.8, 5.9). Figure $5.13(a, b, c)$ show the adaptive moving mesh at times $t = 11$, $t = 20$ and $t = 31$, respectively. We clearly see the mesh adaptation in both x and y directions as the propagating finger gradually develops.

Figure $5.14(a, b)$ illustrate the surface plots (side and top view, respectively) of $h(x, y, t)$ at time $t = 31$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$. We clearly observe the formation of a single finger and visually it appears identical to the uniform mesh solution shown in figure $5.2(a, b)$ (note $M_x = 200$ and $N_y = 200$ for this solution) and $5.6(a, b)$). Figure $5.14(c)$ shows the corresponding adaptive moving mesh and we clearly see clustering of mesh points in the y-direction along the finger.

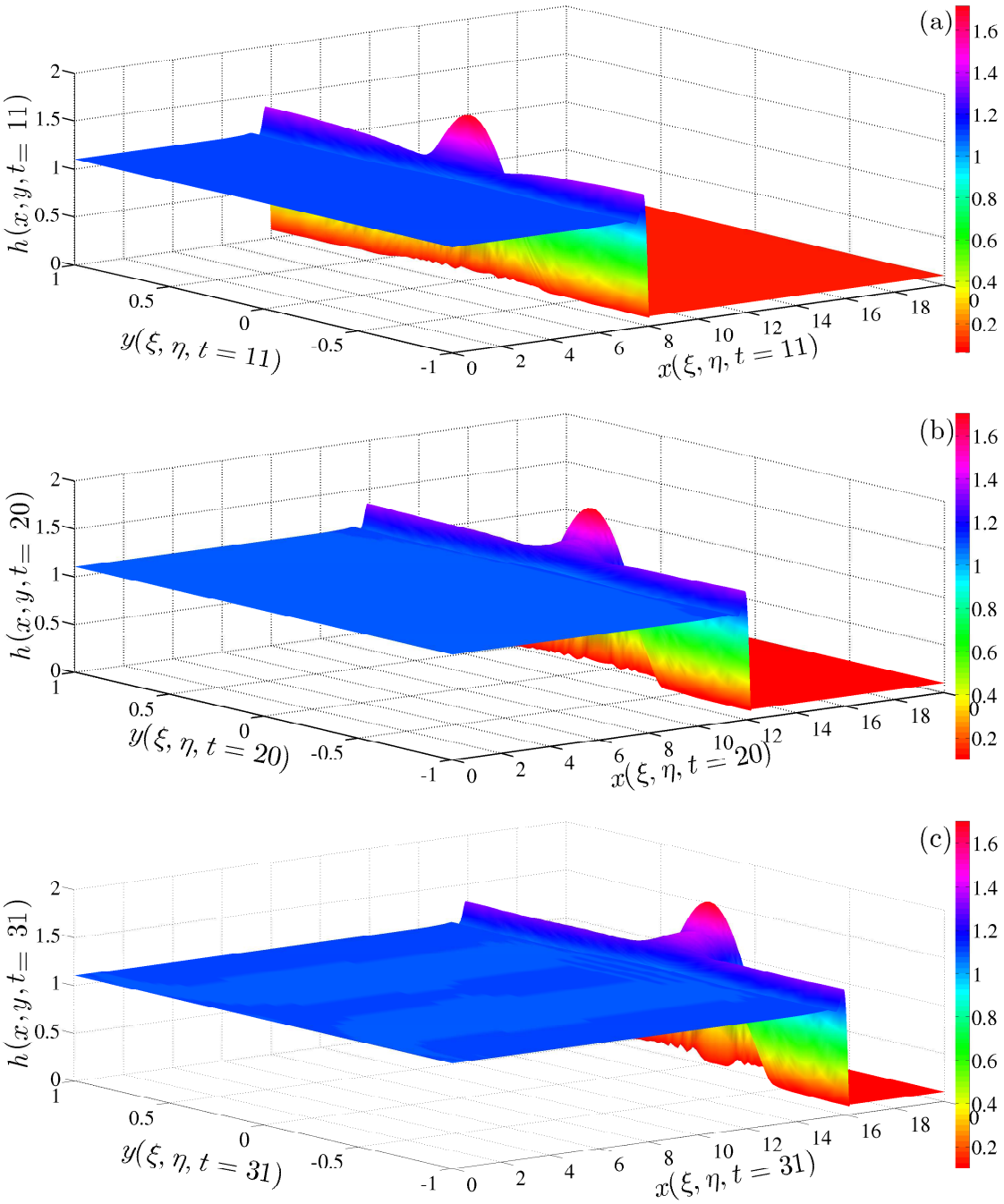

Figure 5.11: Surface plots of $h(x, y, t)$ showing its evolution in time (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$.

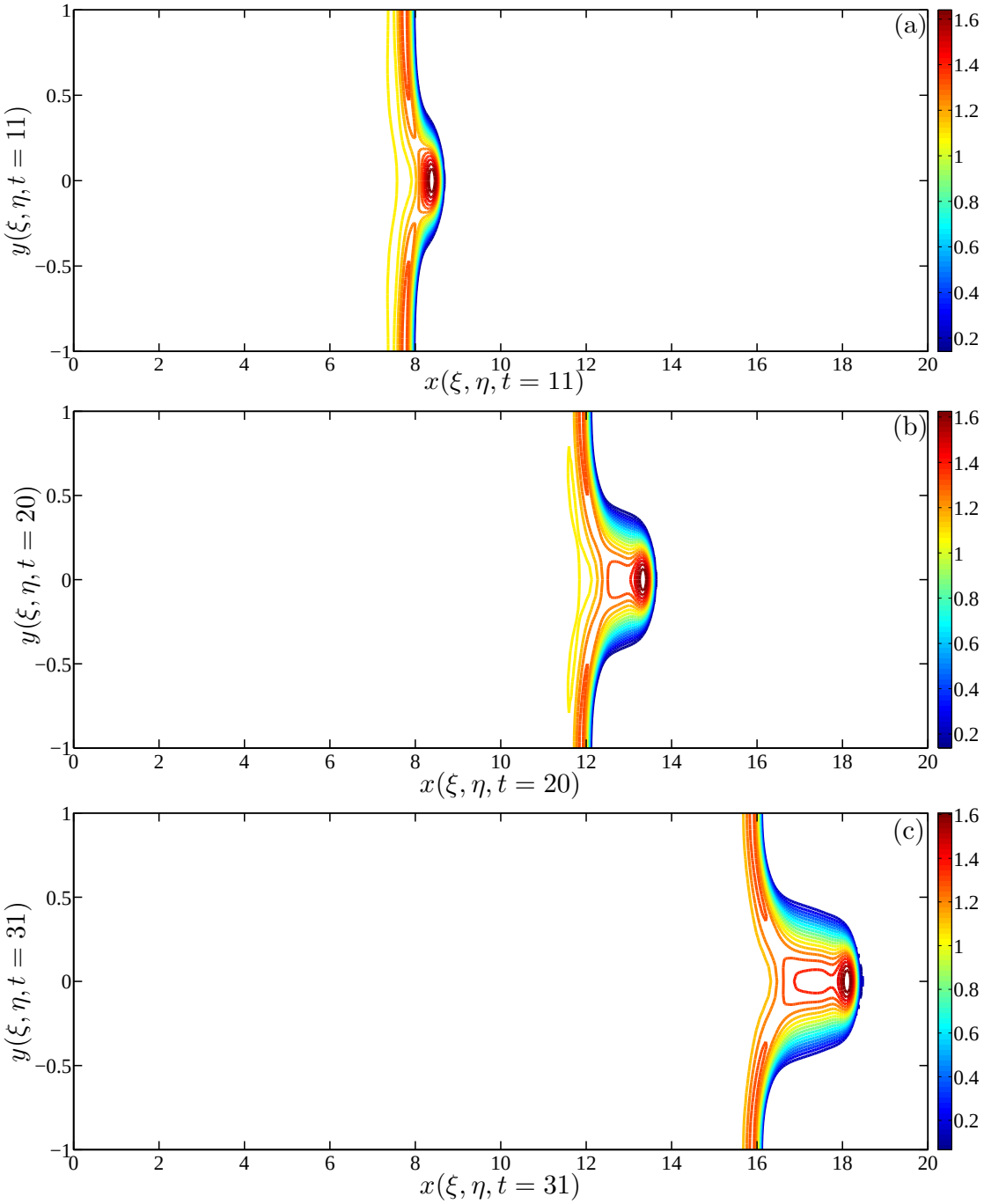

Figure 5.12: Contour plots of $h(x, y, t)$ showing its evolution in time (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$.

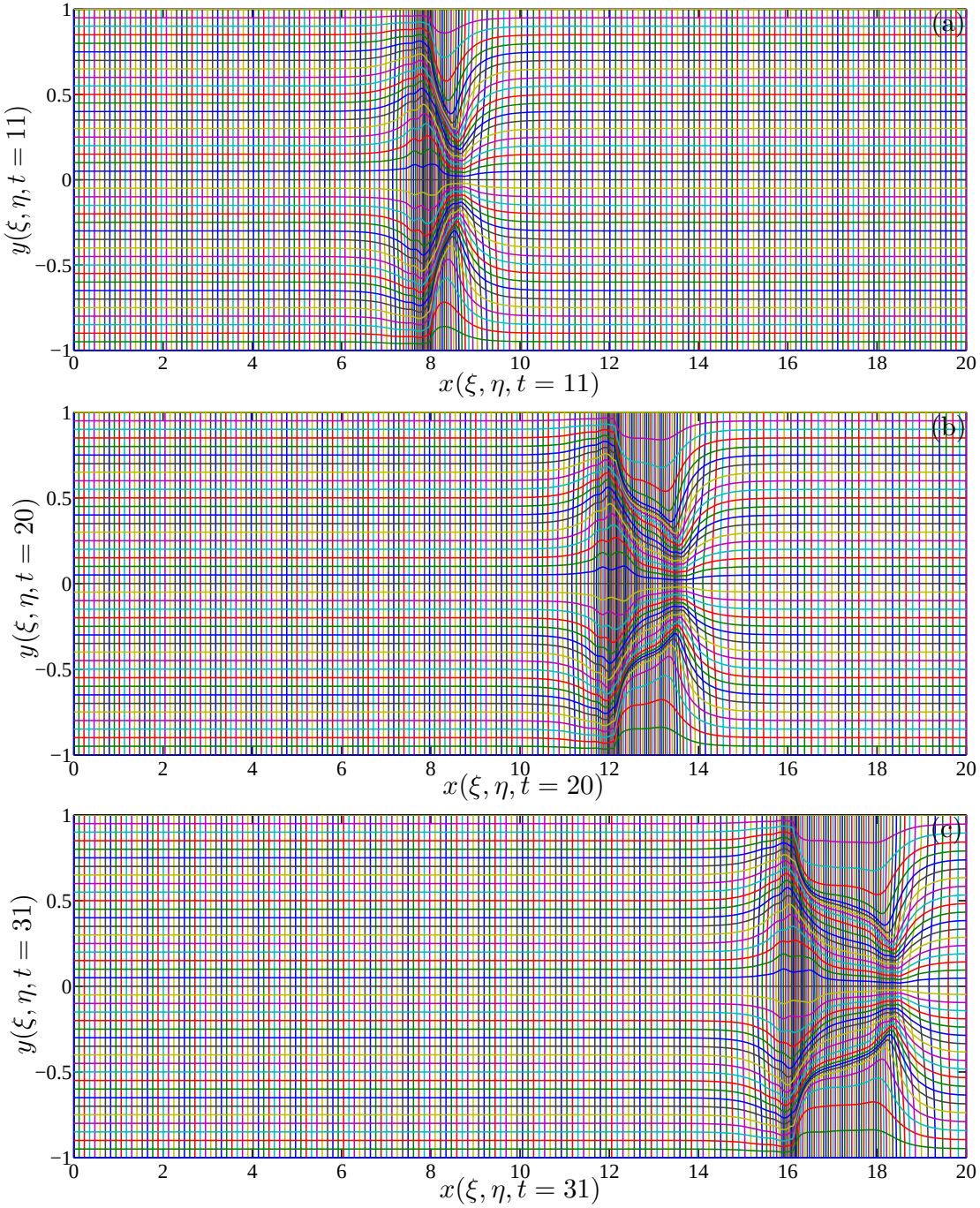

Figure 5.13: The adaptive moving mesh at times (a) $t = 11$, (b) $t = 20$ and (c) $t = 31$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$.

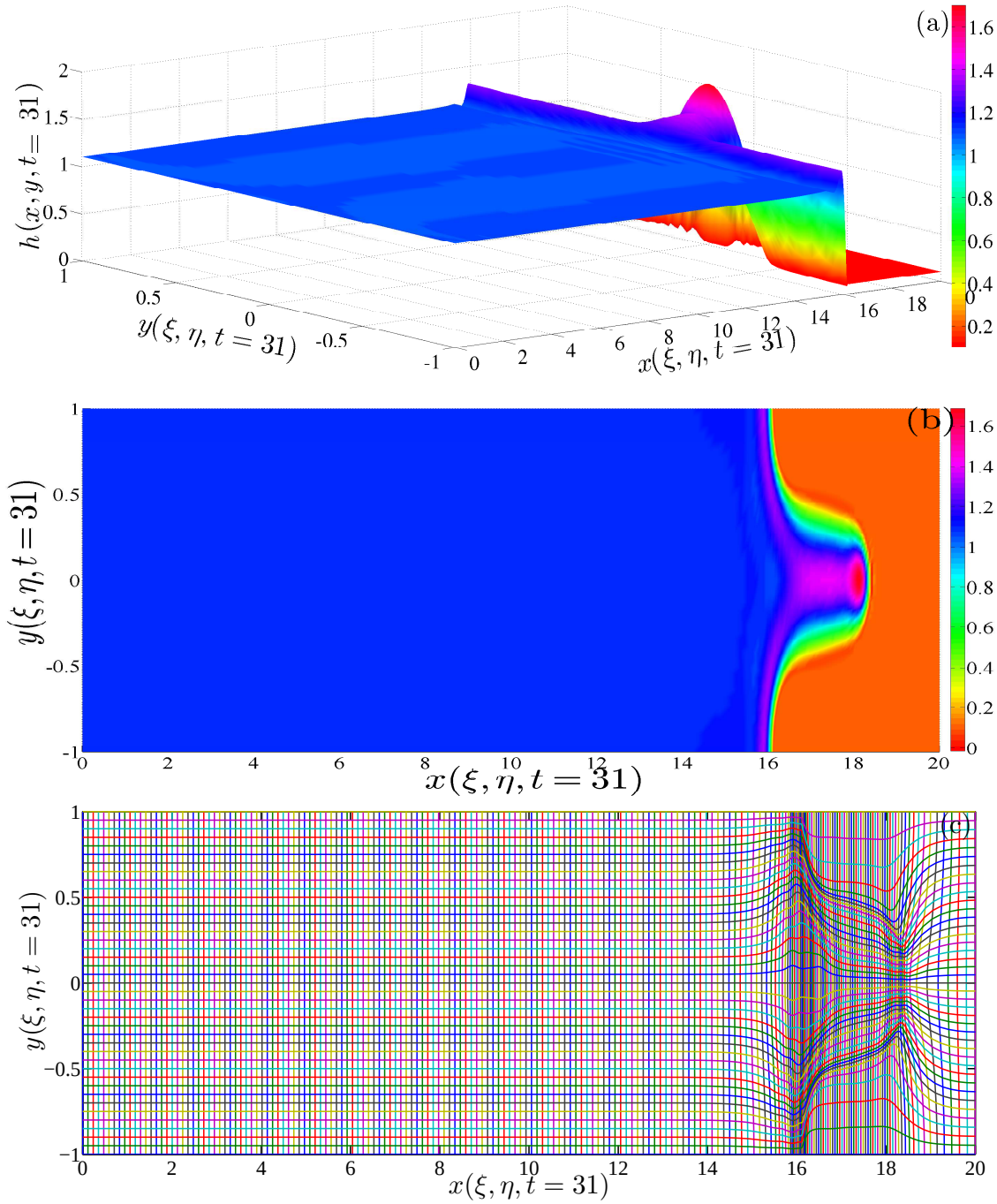

Figure 5.14: Surface plots ((*a*) side view and (*b*) top view) of $h(x, y, t)$ at $t = 31$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$). (*c*) shows the corresponding adaptive moving mesh obtained using the PMA equation and curvature-based monitor function with $\alpha = 1$.

5.2 PDE with constant volume boundary conditions

In this section, we investigate the numerical solution of the spreading of a two-dimensional drop or sheet on both a uniform mesh as well as an adaptive moving mesh. The underlying PDE is given by Eq. (5.1). The physical domain is selected as a rectangle denoted by $-L_x \leq x \leq L_x$ and $-L_y/2 \leq y \leq L_y/2$ which is divided into $(M_x+1) \times (N_y+1)$ mesh points (x_j, y_k) with $j = 1, \ldots, M_x+1$ and $k = 1, \ldots, N_y+1$.

Equation (5.1) is supplemented by the boundary conditions in the x direction, which are

$$h = b, \; h_{xxx} = 0, \quad \text{at} \quad x = \pm L_x, \tag{5.110}$$

and we use symmetry boundary conditions in y direction given by Eq. (5.3). Here, $b \ll 1$, denotes the precursor film thickness. This represents no flux of fluid out of the domain and the drop connects onto a flat precursor film.

As in the previous section, we are interested in the development of the fingering instability for this case starting from a y-independent initial condition with sinusoidal perturbations imposed on it. Based on this, the initial condition is chosen as:

$$\begin{aligned} h(x,y,0) &= h_0(x) + \sum_{l=1,k=1}^{m,n} a_k \cos(k\pi y)\; e^{\left[-K_l\;(x-x_l)^2\right]}, \\ h_0(x) &= (x_0 - x^2)\left[H(x_0-x) - H(-x_0-x)\right] + b\left[H(x-x_0) - H(-x_0-x)\right], \end{aligned} \tag{5.111}$$

where, $H(x)$ is the Heaviside function, x_0 is the initial location where the parabolic shaped drop connects to the precursor thickness film both upstream and downstream, k is the wavenumber of each sinusoidal mode with period $= 2/k$ and amplitude a_k, n is the total number of modes imposed, m is the total number of locations $x = x_l$ in the x direction across where the sinusoidal perturbations are applied and K_l controls the width of the localised perturbations at x_l. Figure 5.15 shows the two-dimensional initial condition Eq. (5.111) with a single transverse perturbation ($k = n = 1$) applied at two locations: $x_1 = 0$ with amplitude $a_1 = 0.1$ and $x_2 = 1$ with amplitude $a_1 = 0.01$. We choose $K_1 = 20$, $K_2 = 100$, the precursor thickness $b = 0.1$, $L_{x_1} = -2$, $L_{x_2} = 18$ and $L_y = 2$.

We seek the 2D evolution of the Eq. (5.1) subject to the BCs given by Eqs. (5.3, 5.110) for the two-dimensional time evolution of the initial condition Eq. (5.111). We first show the numerical solution using a uniform mesh followed by an adaptive mesh. The parameter values used are: $b = 0.1$, $Ca = 10^{-2}$, $\theta = 90^o$ (so, $D(\theta) = 0$), $L_{x_1} = -2$, $L_{x_2} = 18$ and $L_y = 2$.

5.2.1 Numerical solution of Eqs. (5.1, 5.110, 5.3) on a uniform mesh

We use the uniform mesh scheme Eq. (5.7) for discretising Eq. (5.1). As the previous section, fictitious points are used, where necessary, near the ends of the 2D domain and derived from the BCs given by Eqs. (5.110, 5.3).

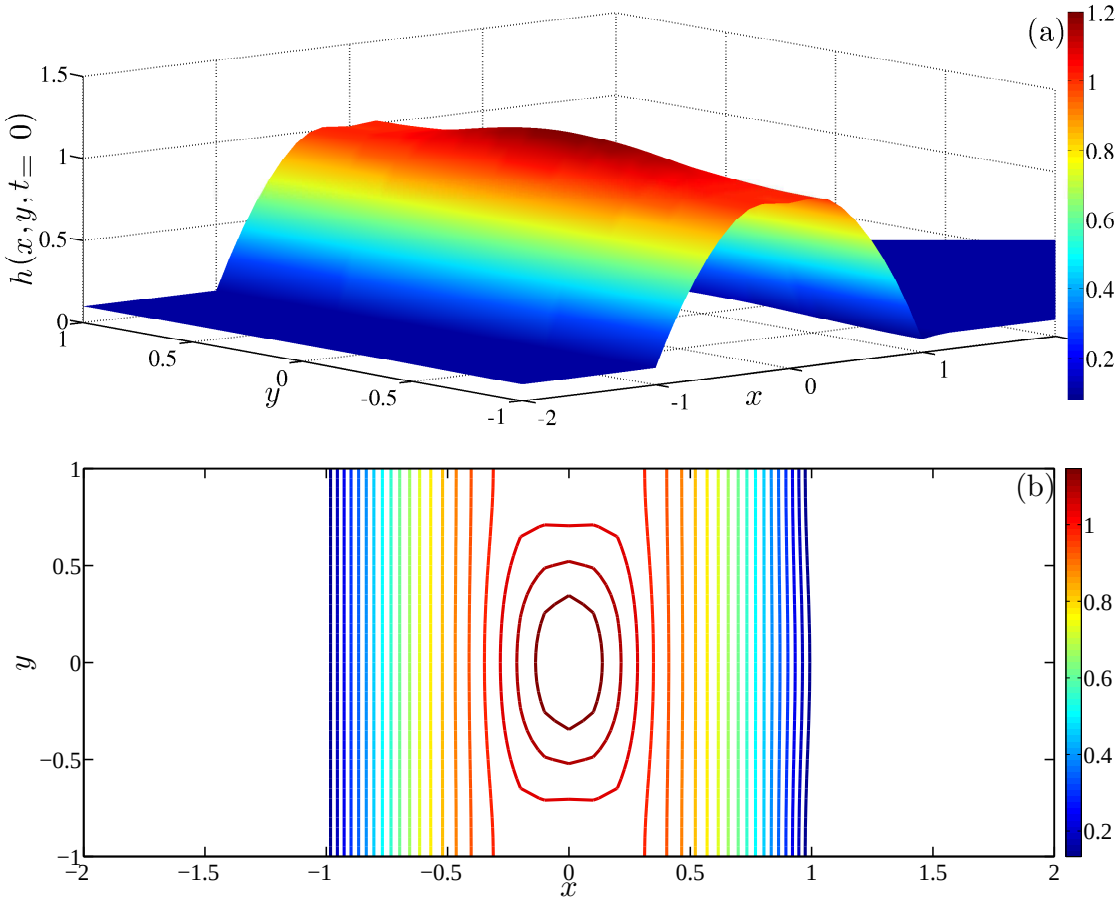

Figure 5.15: The initial condition showing (a) surface and (b) contour plot using Eq. (5.111) with a single transverse perturbation ($k = n = 1$) applied at two locations: $x_1 = 0$ with amplitude $a_1 = 0.1$ and $x_2 = 1$ with amplitude $a_1 = 0.01$. We choose $K_1 = 20$, $K_2 = 100$, the precursor thickness $b = 0.1$, $L_{x_1} = -2$, $L_{x_2} = 18$ and $L_y = 2$.

Figure 5.16(a, b) illustrate the surface plots (side and top view, respectively) of $h(x, y, t)$ at time $t = 80$. We clearly observe the formation of a single finger which is consistent with the linear stability analysis for the parameter values considered here. Figure 5.17(a, b, c) illustrate the surface plots of $h(x, y, t)$ showing its evolution in time (times shown are $t = 10$, $t = 40$ and $t = 80$). At early time $t = 10$, a finger is observed to slowly start forming. As time t increases, the finger appears to develop with a preferred width which is consistent with the linear stability analysis. This is more clearly depicted in the contour plots presented in figure 5.16(a, b, c) where the width of the finger is approximately one and it lengthens as time t increases.

5.2.2 Numerical solution of Eqs. (5.1, 5.110, 5.3) on an adaptive mesh

We use the adaptive moving mesh scheme Eq. (5.29) for discretising Eq. (5.1) with MMPDE4 given by the discretisation in Eq. (5.49) and the curvature-based monitor function discretisation given by Eq. (5.51).

Figure 5.19(a, b, c) illustrate the surface plots of $h(x, y, t)$ showing its evolution in time (times shown are $t = 20$, $t = 40$ and $t = 80$) using the adaptive moving mesh scheme with $M_x = 200$ and $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature monitor function with $\alpha = 1$. At early time $t = 10$, a finger is observed to slowly start forming. As time t increases, the finger appears to develop with a preferred width. This is more clearly depicted in the contour plots presented in figure

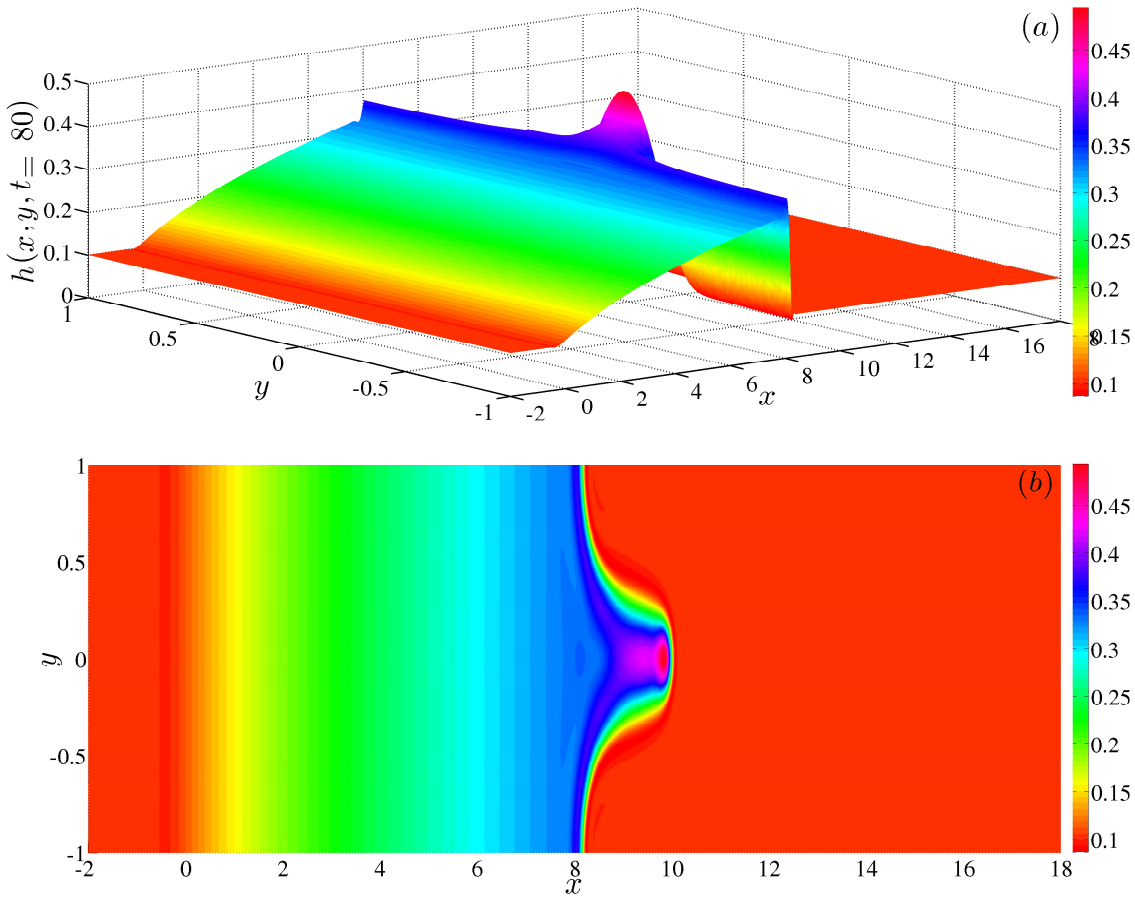

Figure 5.16: Surface plot ((*a*) side view and (*b*) top view) of $h(x, y, t)$ at $t = 80$. We clearly observe the formation of a fingering instability. The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_{x_1} = -2$, $L_{x_2} = 18$, $L_y = 2$, $M_x = 2000$ (so, $\Delta x = 0.01$) and $N_y = 200$ (so, $\Delta y = 0.01$).

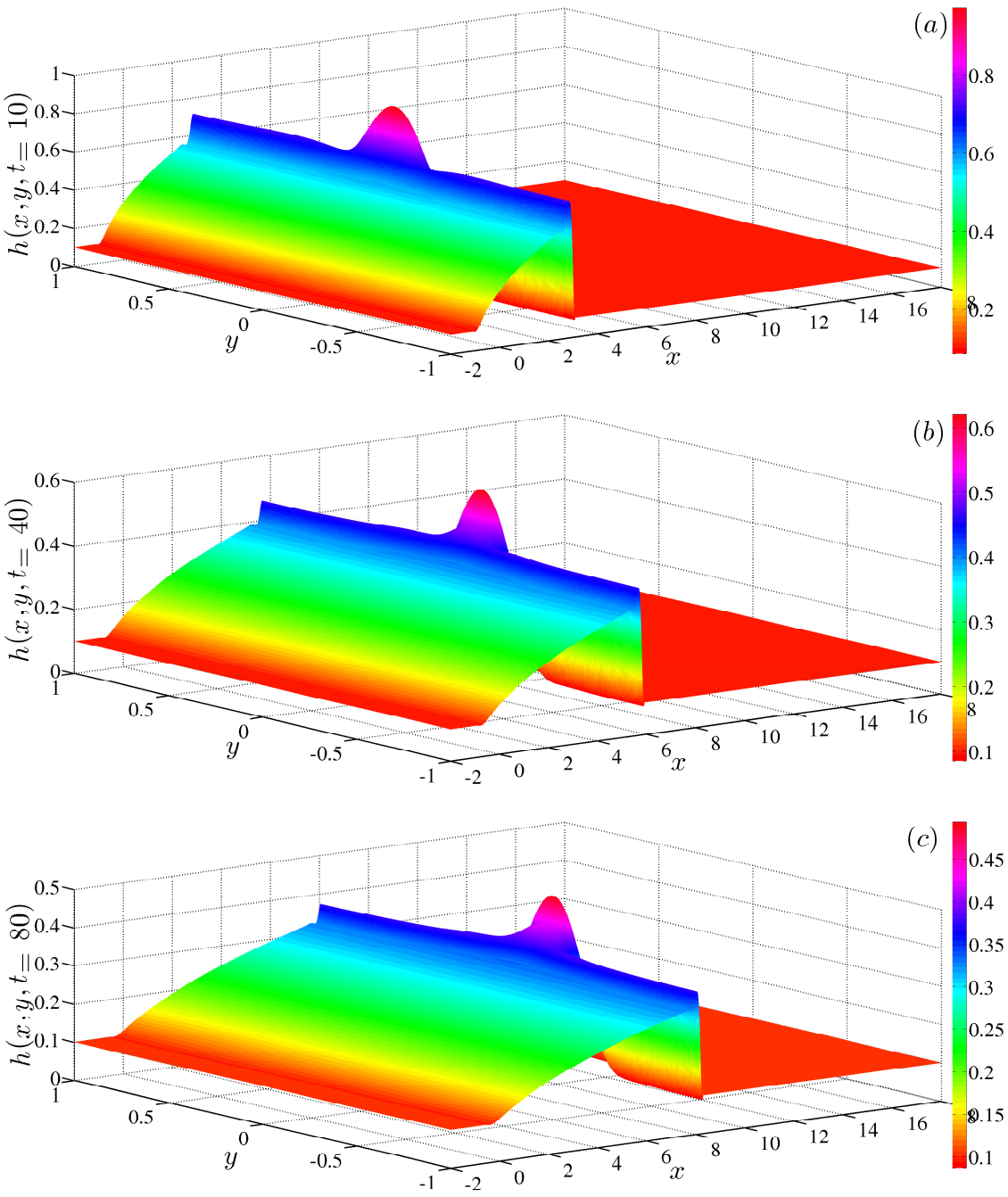

Figure 5.17: Surface plots of $h(x, y, t)$ showing its evolution in time (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$). The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_{x_1} = -2$, $L_{x_2} = 18$, $L_y = 2$, $M_x = 2000$ (so, $\Delta x = 0.01$) and $N_y = 200$ (so, $\Delta y = 0.01$).

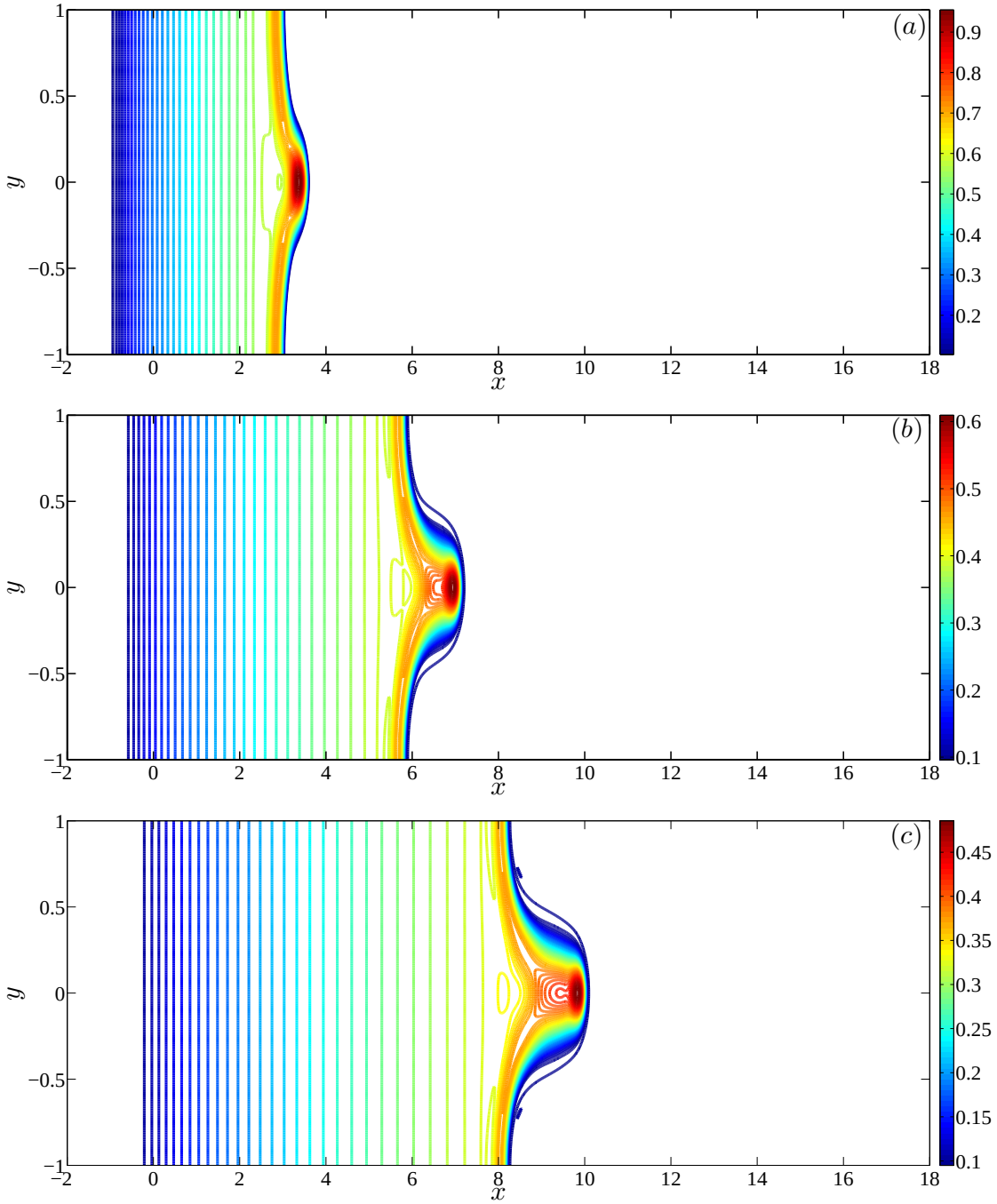

Figure 5.18: Contour plots of $h(x, y, t)$ showing its evolution in time (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$). The parameter values are: $Ca = 10^{-3}$, $\theta = 90^o$ (so, $D = 0$), $b = 10^{-1}$, $L_{x_1} = -2$, $L_{x_2} = 18$, $L_y = 2$, $M_x = 2000$ (so, $\Delta x = 0.01$) and $N_y = 200$ (so, $\Delta y = 0.01$).

5.20(a, b, c) where the width of the finger is approximately one and it lengthens as time t increases. These results are indistinguishable from those obtained using a uniform mesh (see figures 5.17, 5.18). Figure 5.21(a, b, c) show the adaptive moving mesh at times $t = 20$, $t = 40$ and $t = 80$, respectively. We clearly see the mesh adaptation in both x and y as the propagating finger gradually develops. Figure 5.22(a, b) illustrate the surface plots (side and top view, respectively) of $h(x, y, t)$ at time $t = 80$. We clearly observe the formation of a single finger and visually it appears identical to the uniform mesh solution shown in figure 5.16(a, b) (note $M_x = 2000$ and $N_y = 200$ for this solution). Figure 5.22(c) shows the corresponding adaptive moving mesh and we clearly see clustering of mesh points in the y-direction along the finger.

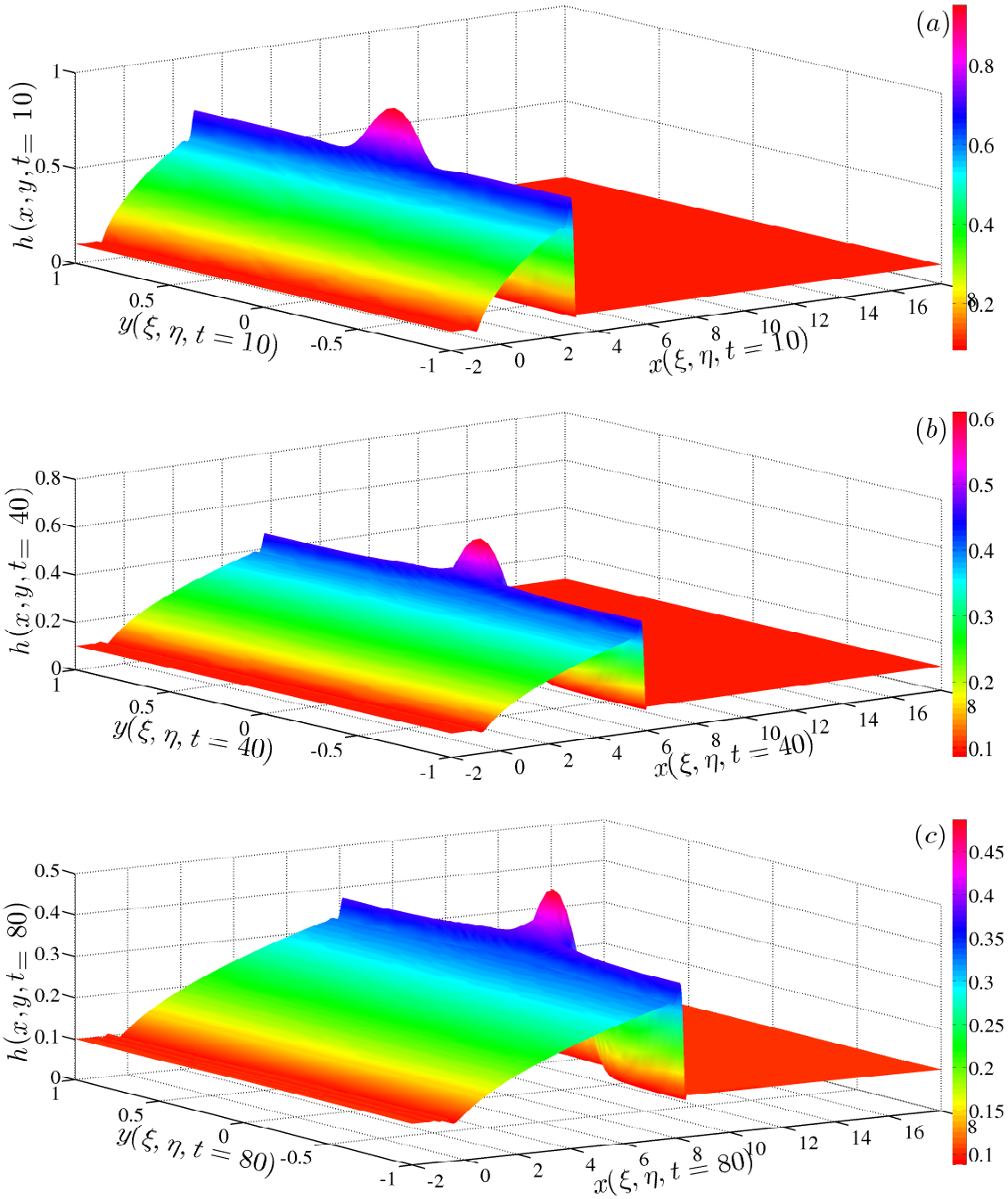

Figure 5.19: Surface plots of $h(x, y, t)$ showing its evolution in time (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

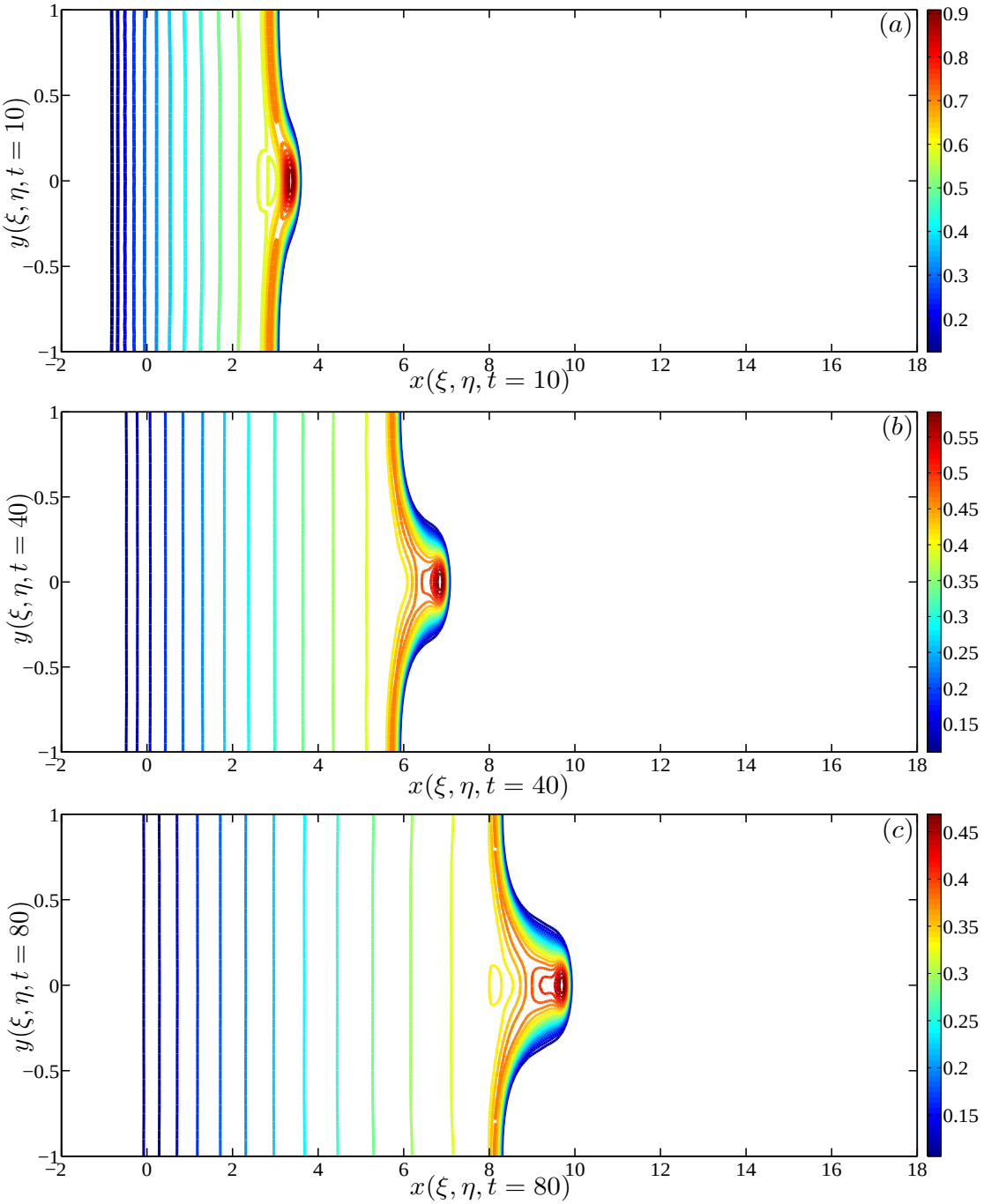

Figure 5.20: Contour plots of $h(x, y, t)$ showing its evolution in time (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

5.2.3 Numerical solution of Eqs. (5.1,5.2,5.3) using the Parabolic-Mongè-Ampere (PMA) equation

We use the adaptive moving mesh scheme Eq. (5.69) for discretising Eq. (5.1) with the PMA equation given by the discretisation in Eq. (5.100) and the curvature-based monitor function discretisation given by Eq. (5.51).

Figure 5.23(a, b, c) illustrate the surface plots of $h(x, y, t)$ showing its evolution in time (times shown are $t = 20$, $t = 40$ and $t = 80$). These numerical solutions are obtained using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$. At early time $t = 10$, a finger is observed to slowly start forming. As time t increases, the finger appears to develop with a preferred width. This is more clearly depicted in the contour plots presented in figure 5.24(a, b, c) where the width of the finger is approximately one and it lengthens as time t increases. These results are similar to those obtained using both a uniform mesh (see figures 5.17, 5.18) and adaptive moving mesh using MMPDE4 (see 5.19, 5.20). Figure 5.25(a, b, c) show the adaptive moving mesh at times $t = 20$, $t = 40$ and $t = 80$, respectively. We clearly see the mesh adaptation in both x and y as the propagating finger gradually develops. Figure 5.26(a, b) illustrate the surface plots (side and top view, respectively) of $h(x, y, t)$ at time $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$. We clearly observe the formation of a single finger and visually it appears

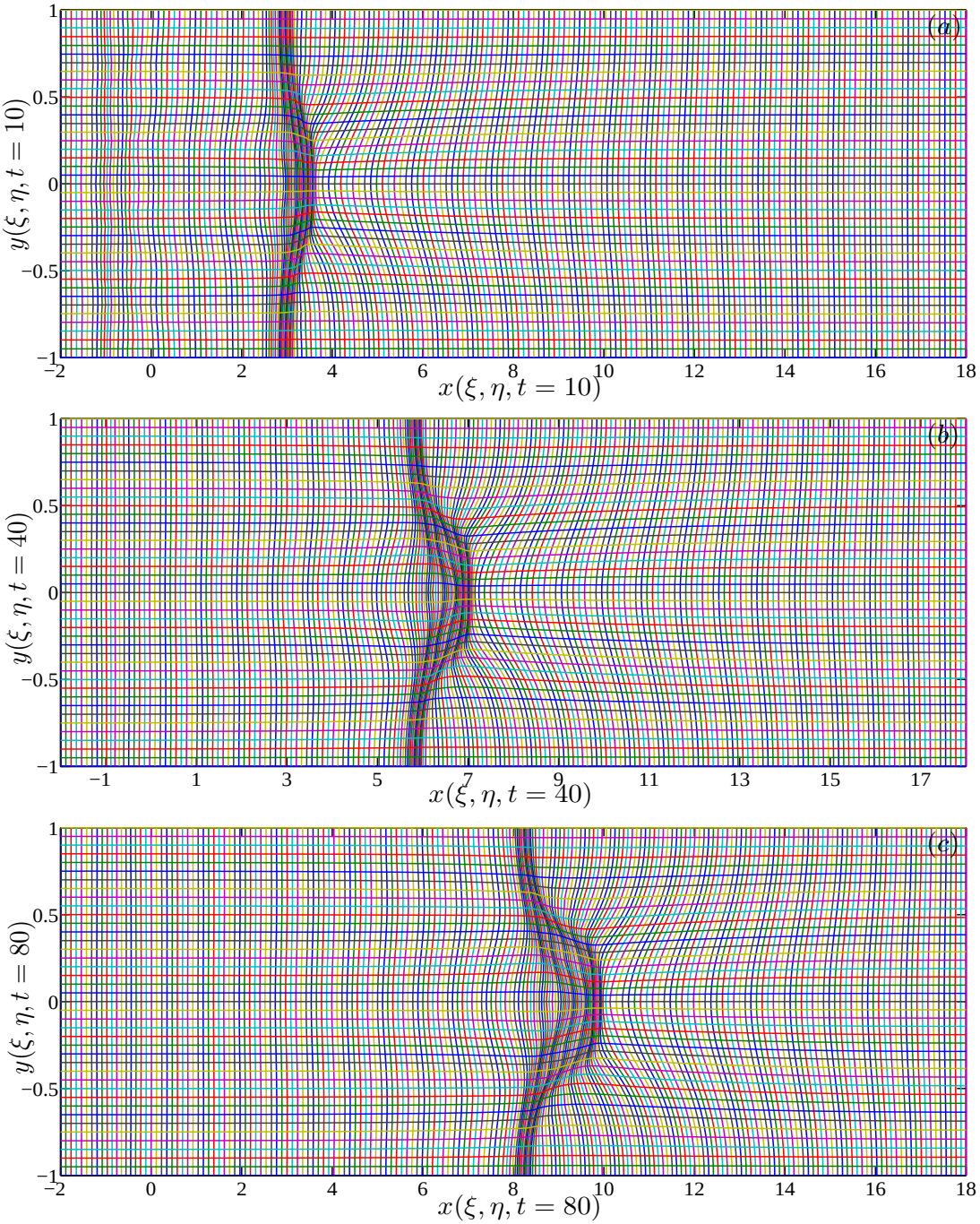

Figure 5.21: The adaptive moving mesh at time (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

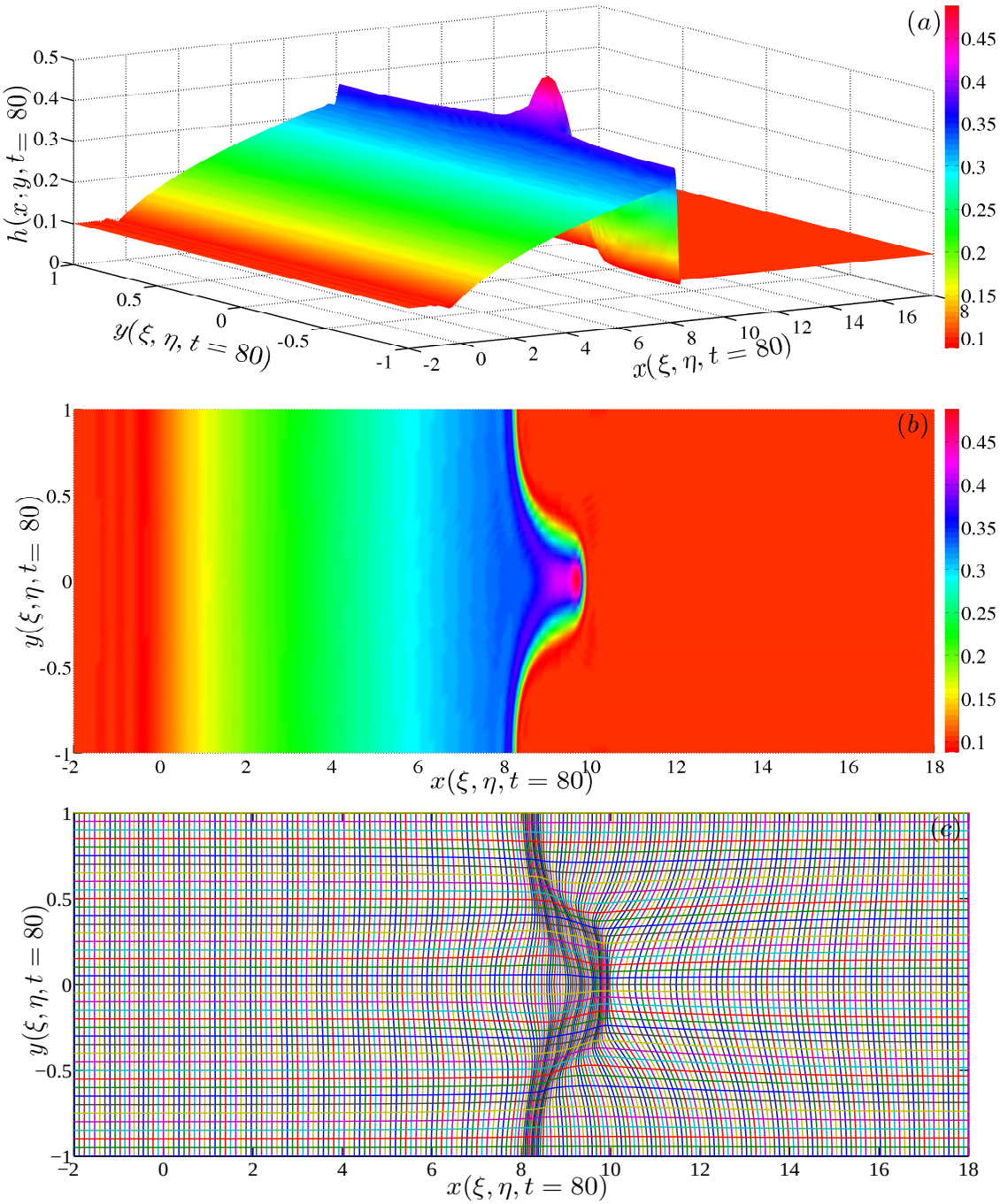

Figure 5.22: Surface plots ((*a*) side view and (*b*) top view) of $h(x, y, t)$ at $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$). (*c*) shows the corresponding adaptive moving mesh obtained using MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$.

identical to the uniform mesh solution shown in figure 5.16(a, b) (note $M_x = 200$ and $N_y = 200$ for this solution) and 5.22(a, b)). Figure 5.26(c) shows the corresponding adaptive moving mesh and we clearly see clustering of mesh points in the y-direction along the finger.

5.3 Conclusions

In this chapter we presented a robust and accurate finite difference method on an adaptive moving mesh for the two-dimensional gravity-driven spreading of a liquid sheet or drop down an inclined plane. The adaptive moving mesh strategy was based on an MMPDE (namely, MMPDE4) and an optimal transport equation, the PMA equation. By comparing the results using the MMPDE4 and PMA equations with those from a uniform mesh method, it was shown that the adaptive moving mesh methods are accurate and offer significant reductions in memory requirements (for example, a converged uniform mesh solution required 2000×200 mesh points in the x and y directions, respectively, while the adaptive moving mesh schemes required much less mesh points 200×40). The CPU times, however, for the MMPDE4 adaptive moving mesh solution corresponding to 200×200 mesh points was comparable to the uniform mesh solution with 2000×200 mesh points. This is due to the additional two mesh evolution PDEs that are also required to be solved simultaneously with the underlying PDE. The PMA adaptive moving mesh solution took less CPU time compared to both MMPDE4 and the uniform mesh solution because there were only two coupled PDEs to

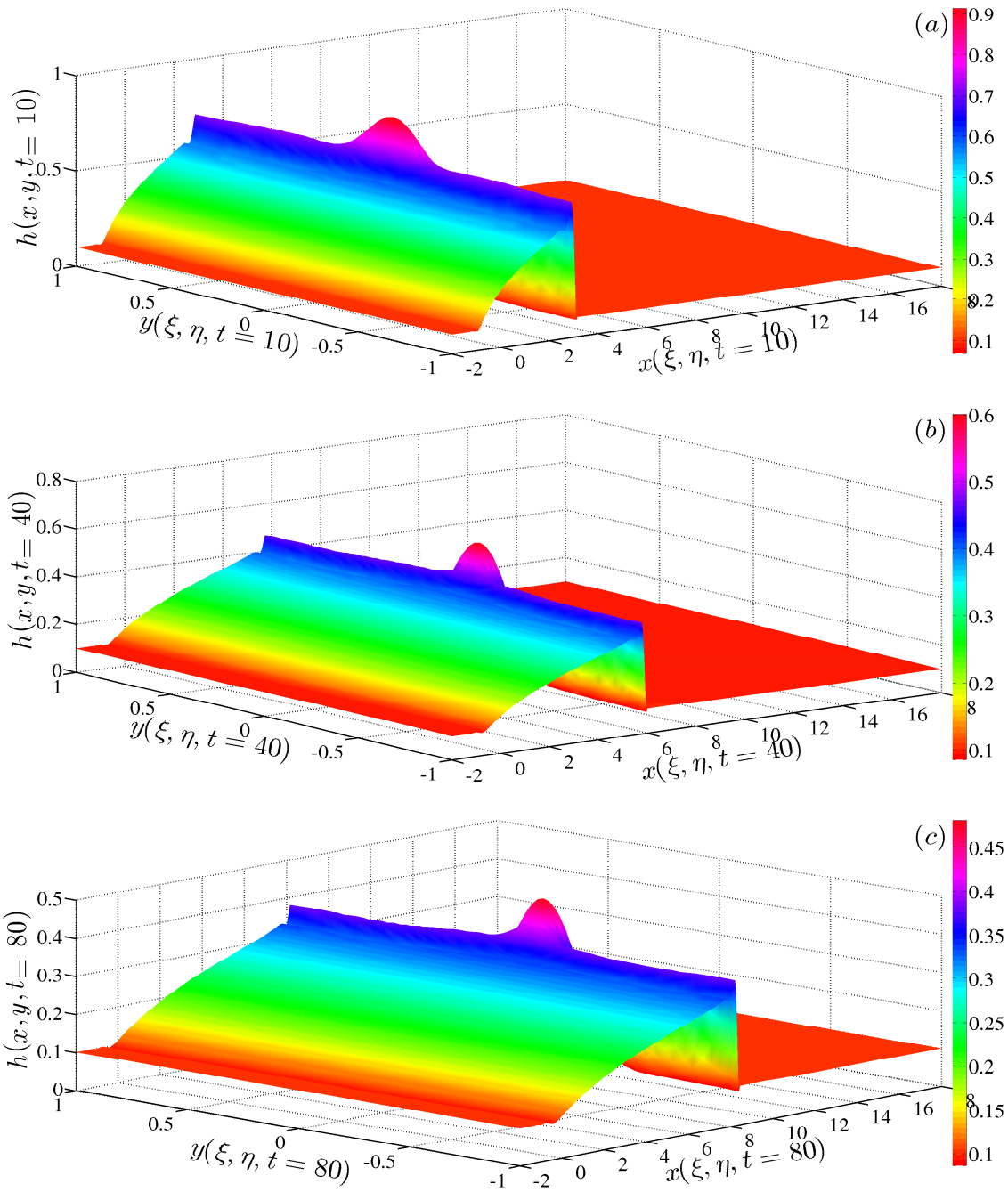

Figure 5.23: Surface plots of $h(x, y, t)$ showing its evolution in time (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$.

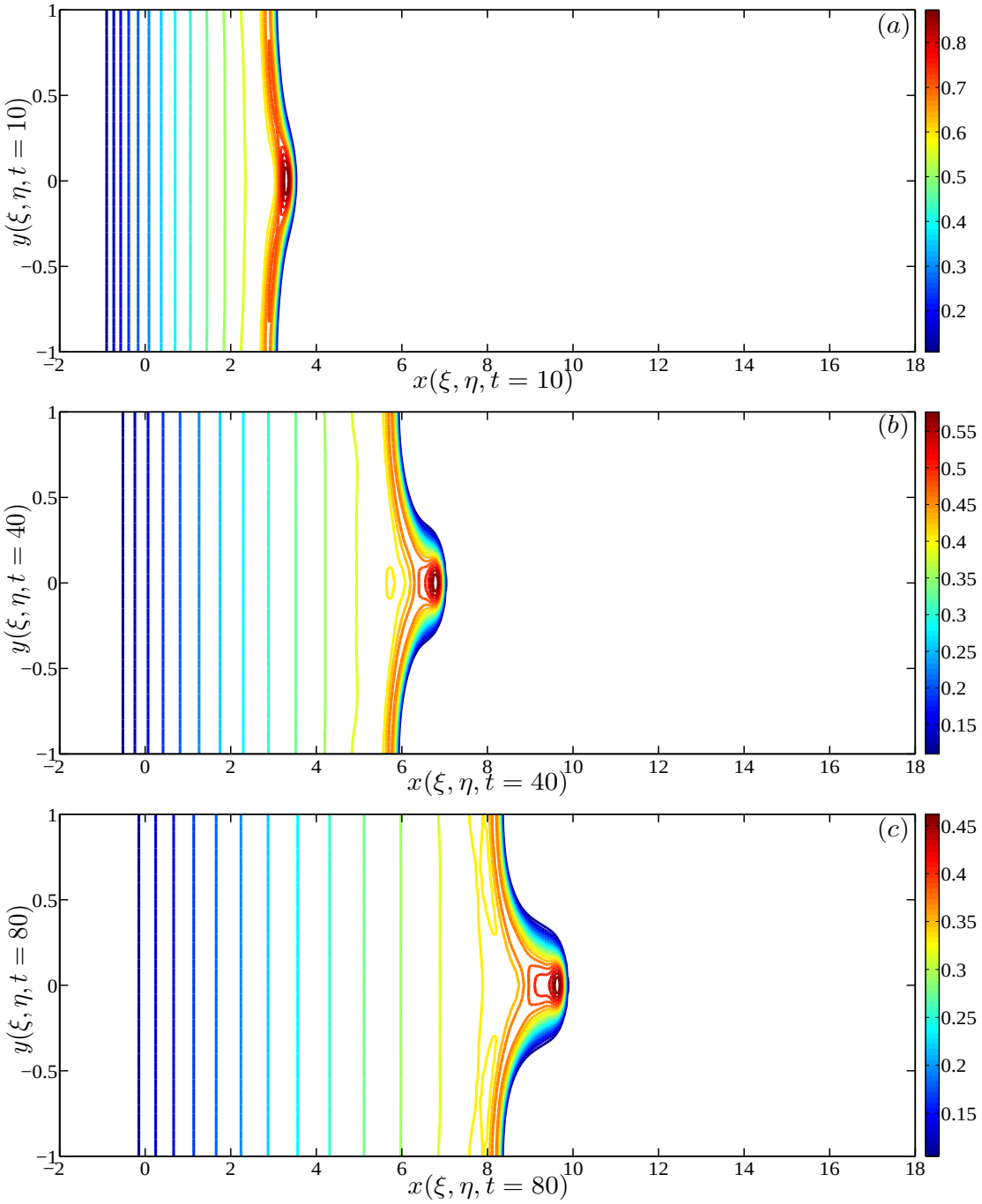

Figure 5.24: Contour plots of $h(x, y, t)$ showing its evolution in time (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$.

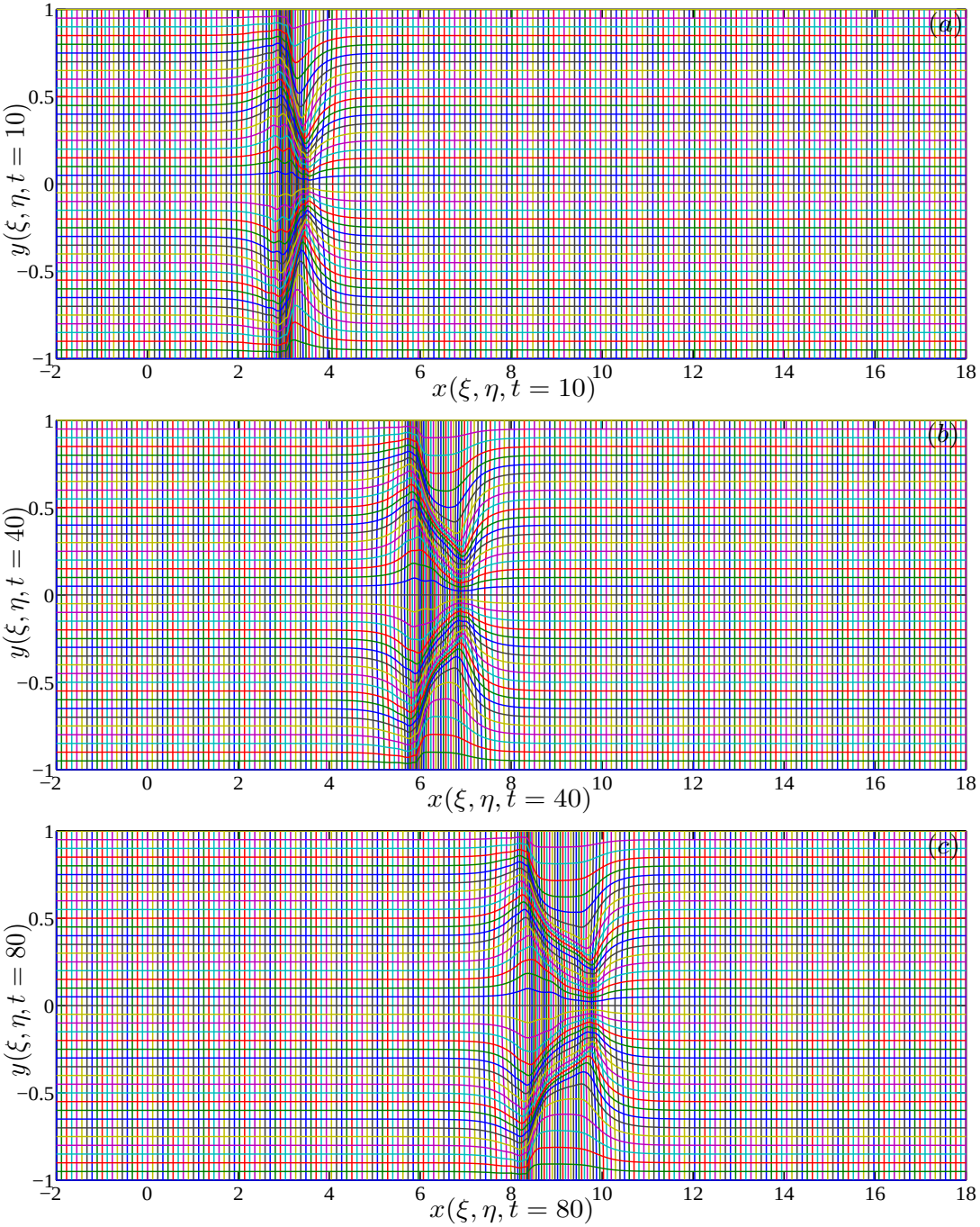

Figure 5.25: The moving adaptive mesh at times (a) $t = 10$, (b) $t = 40$ and (c) $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$), PMA equation and curvature-based monitor function with $\alpha = 1$.

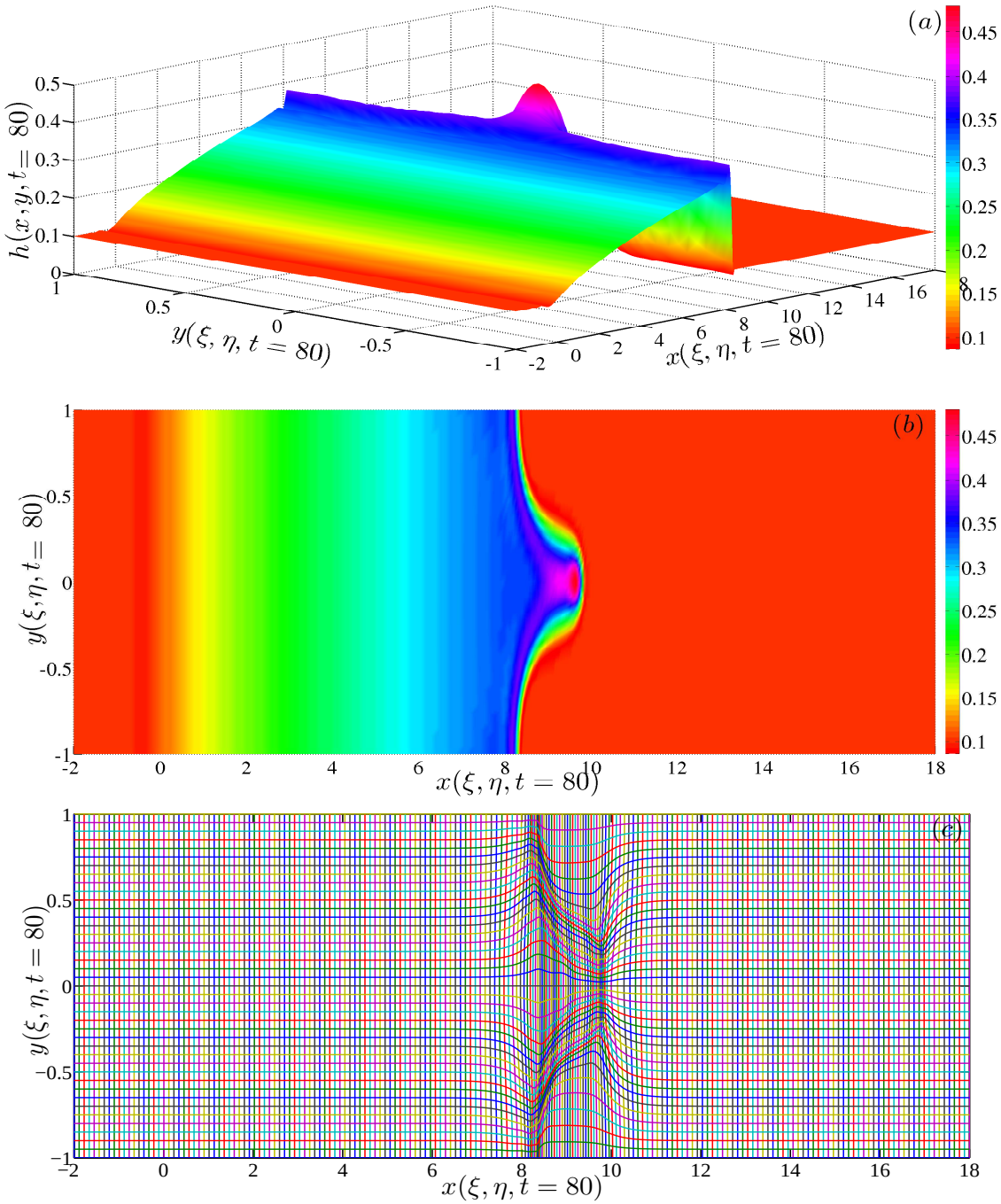

Figure 5.26: Surface plots ((*a*) side view and (*b*) top view) of $h(x, y, t)$ at $t = 80$ using the adaptive moving mesh scheme with $M_x = 200$, $N_y = 40$ (so, initial $\Delta x = 0.1$ and $\Delta y = 0.05$). (*c*) shows the corresponding adaptive moving mesh obtained using the PMA equation and curvature-based monitor function with $\alpha = 1$.

solve. As mentioned in Chapter 1, the strategy of solving the PDE and mesh equations alternately may reduce the CPU times further.

While the emphasis here was in simulating the fingering instability for a single finger, we can modify the symmetry BCs on the y-boundaries (both for the underlying PDE and mesh equations) to incorporate periodic BCs. This would make the bandwidth of the system larger and may increase CPU times but would enable us to simulate multiple non-symmetric fingers. This will be explored in future. We have not made any direct comparisons on accuracy and CPU times with corresponding hp-refinement techniques used by Sun *et al.* [77] and Li *et al.* [62]. However, if we visually compare the adapted mesh from our solution (see figures 5.10,5.21) to figures 6, 7 in Sun *et al.* [77] and figure 4 in Li *et al.* [62], we see a very close resemblance. This suggests that the adaptive schemes are almost similar in their accuracy but their efficiency in their performance needs further investigation.

In conclusion, our results indicate great promise for MMPDEs and the PMA equation in two-dimensional thin-film flow problem. Although we have only considered a specific form of the underlying PDE, the general framework presented in this chapter can be utilised for any two-dimensional thin-film equation of the form considered here.

Chapter 6

Future Work

We have successfully applied the r-adaptive moving mesh method based on moving mesh PDEs (MMPDEs) for two prototype one and two-dimensional thin film equations considered in this thesis. Our results indicate great promise in terms of simplicity in its implementation and efficiency for this method to be applied on a regular basis to thin film equations, in general. However, there are challenging problems in the thin film literature where this method needs to be further tested before its success can be guaranteed. We list some of these challenges which will be considered in the future.

1. Drop spreading with a dynamic (or moving) contact angle. The spreading problems considered in this thesis assume that the plane or substrate is pre-wetted with a thin precursor film of thickness b. This is a mathematical regularisation required to remove the so-called contact line singularity when a no-slip boundary condition is applied at the substrate. Another regularisation is to introduce some

amount of slip which along with a moving contact line condition allows studying spreading problems with dynamic (or moving) contact lines (rather than the effective contact line considered here). In the gravity-driven spreading problem considered in Chapters 3 and 5 this would result in much steeper slopes around the dynamic contact line (analogous to the case when $b \ll 1$). Adapting the mesh around the this region to accurately resolve the solution there using the r-adaptive method will be considered in future.

[2] Spreading of a surfactant-laden liquid drop on a horizontal pre-wetted plane. As was discussed in Chapter 1, this problem displays intricate wave-like structures and development of ultra-thin films for the one-dimensional problem (see figure 6.1 and Jensen & Naire [55] for the one-dimensional numerical solution description) and a dendritic (or branching) fingering instability for the two-dimensional case (see figure 6.2 (a, b), Warner *et al.* [83, 84] and Edmonstone *et al.* [33]). The one-dimensional numerical solutions by Jensen & Naire [55] were obtained using a nonuniform (but fixed) mesh manually clustering mesh points in the regions of rapid variation in the solution. We are interested in applying the r-adaptive method to redistribute the mesh points automatically to these regions based on a suitable monitor function. The framework using MMPDEs of the type discussed in this thesis will be directly applicable to this problem but the main challenge will be in determining an appropriate monitor function. Warner *et al.* [83, 84] obtained the fingering solution shown in figure 6.2 using finite differences on a

uniform mesh using the ADI method. We plan to use the r-adaptive method to obtain this solution in future.

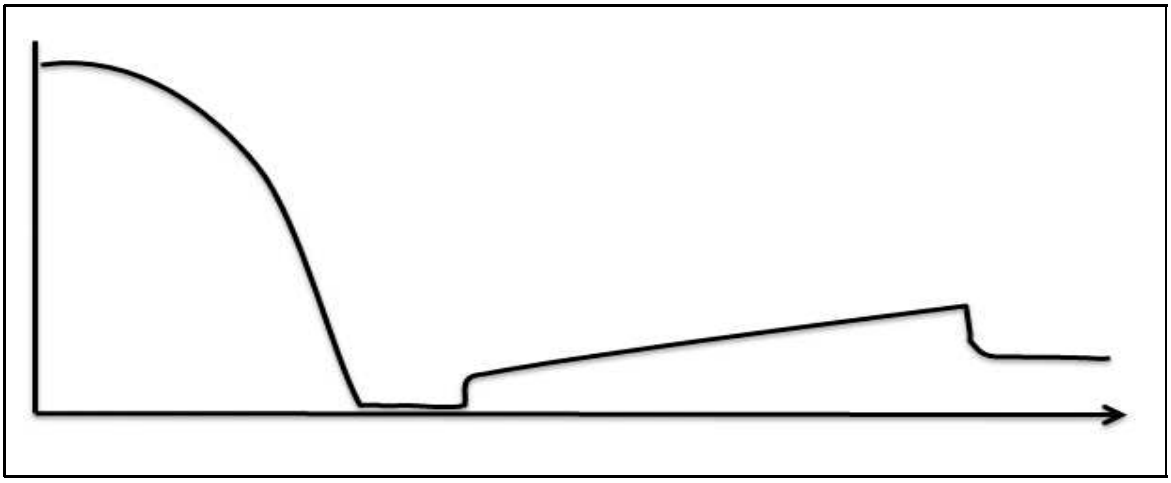

Figure 6.1: This figure illustrates the one-dimensional solution for a surfactant-laden drop spreading on horizontal pre-wetted plane.

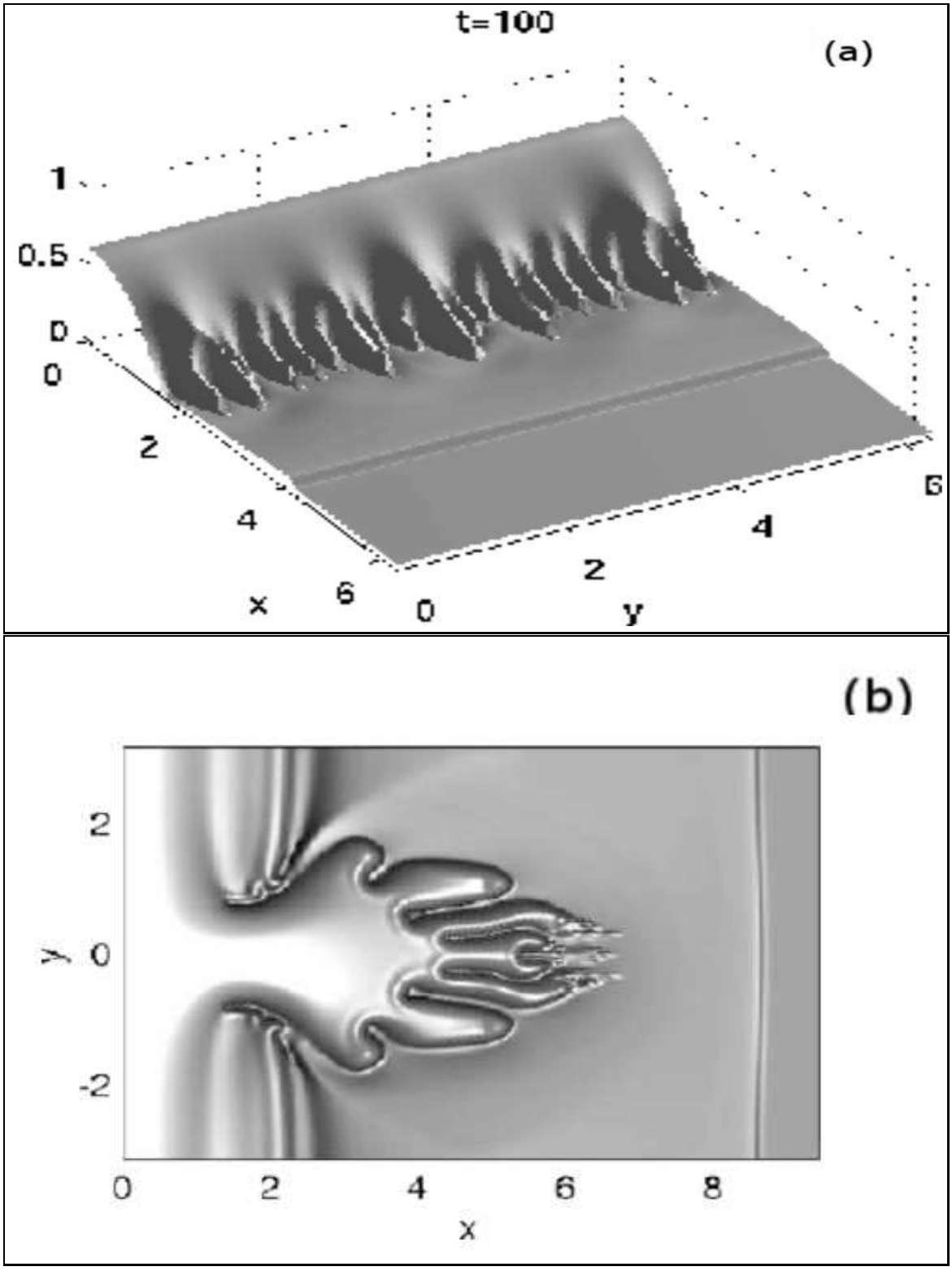

Figure 6.2: Numerical solution of surfactant-induced fingering instability. (a) shows fingering-type pattern from Warner *et al.* [84, 83] and (b) shows tip splitting from Edmonstone *et al.* [33].

Bibliography

[1] A.B. Afsar-Siddiqui, P.F. Luckham, and O.K. Matar. Unstable spreading of aqueous anionic surfactant solutions on liquid films. 2. highly soluble surfactant. *Langmuir*, 19:703–708, 2003.

[2] C. Ancey. Plasticity and geophysical ows: a review. *J. Non-Newt Fluid Mech.*, 142:4–35, 2007.

[3] Jr. Andrew B. White. On selection of equidistributing meshes for two-point boundary-value problems. *SIAM Journal on Numerical Analysis*, 16:472502, 1979.

[4] J. Barrett, J.F. Blowey, and H. Garcke. Finite element approximation of a fourth order degenerate parabolic equation. *Numer. Math.*, 80:525–556, 1998.

[5] G. Beckett and J.A. Mackenzie. Convergence analysis of finite difference approximations on equidistributed grids to a singularly perturbed boundary value problem. *Applied Numerical Mathematics*, 35:87–109, 2000.

[6] A. L. Bertozzi. The mathematics of moving contact lines in thin liquid films. *Notices Amer. Math. Soc.*, 45(6):689 – 697, 1998.

[7] A. L. Bertozzi and M. P. Brenner. Linear stability and transient growth in driven contact lines. *Phys. Fluids*, 9:530–539, 1997.

[8] A.L. Bertozzi and M.P. Brenner. Linear stability and transient growth in driven contact lines. *Phys. Fluids*, 9(3):530–539, 1997.

[9] J.G. Blom and P.A. Zegeling. Algorithm 731: A moving-grid interface for systems of one-dimensional partial differential equations. *ACM Trans. Math. Softw.*, 20:194–214, 1994.

[10] J.U. Brackbill. An adaptive grid with directional control. *J. Computational Phys.*, 108:38–50, 1993.

[11] Yann Brenier. Polar factorization and monotone rearrangement of vector-valued functions. *Communications on Pure and Applied Mathematics*, 44:375–417, 1991.

[12] P.N. Brown, A.C. Hindmarsh, and L.E. Petzold. Using krylov methods in the solution of large-scale differential- algebraic systems. *SIAM J. Sci. Comput.*, 15:1467–1488., 1994.

[13] C. J. Budd, W. Huang, and R. D. Russell. Adaptivity with moving grids. *Acta Numerica*, 18:111–241, 2009.

[14] C. J. Budd and J. F. Williams. Parabolic monge-ampère methods for blow-up problems in several spatial dimensions. *J. Phys. A: Math. Gen.*, 39(19):5425–5444, 2006.

[15] C. J. Budd and J. F. Williams. Moving mesh generation using the parabolic mongeampre equation. *SIAM J. Sci. Comput*, 31:3438–3465, 2009.

[16] C. J. Budd and J. F. Williams. Moving mesh generation using the parabolic mongeampre equation. *SIAM journal on scientic computing*, 31:3438–3465, 2009.

[17] C.J. Budd, M.J.P. Cullen, and E.J. Walsh. Monge-ampère based moving mesh methods for numerical weather prediction, with applications to the eady problem. *Journal of Computational Physics*, 236:247–270, 2013.

[18] H. G. Burchard. Splines (with optimal knots) are better. *Appl. Anal.*, 3:309319, 1974.

[19] L. A. Caffarelli. Boundary regularity of maps with convex potentials. *Communications on Pure and Applied Mathematics*, 45:1141–1151, 1992.

[20] W. Cao, W. Huang, and R. D. Russell. A study of monitor functions for two-dimensional adaptive mesh generation. *SIAM J. Sci. Comput.*, 20:1978–1994, 1999.

[21] W. Cao, W. Huang, and R.D. Russell. An r-adaptive finite element method based upon moving mesh pdes. *Journal of Computational Physics*, 149:221244, 1999.

[22] W. Cao, W. Huang, and R.D. Russell. A moving mesh method based on the geometric conservation law. *SIAM J. Sci. Comput*, 24:118–142, 2002.

[23] W. Cao, W. Huang, and R.D. Russell. Approaches for generating moving adaptive

meshes: location versus velocity. *Applied Numerical Mathematics*, 47:121–138, 2003.

[24] H. D. Ceniceros and T. Y. Hou. An efcient dynamically adaptive mesh for potentially singular solutions. *Journal of Computational Physics*, 172:609–639, 2001.

[25] H. D. Cenicerosa and T. Y. Hou. An effcient dynamically adaptive mesh for potentially singular solutions. *Journal of Computational Physics*, 172:609639, 2001.

[26] R.V. Craster and O.K. Matar. Dynamics and stability of thin liquid films. *Rev. Mod. Phys.*, 81:1131–1198, 2009.

[27] C. de Boor. Good approximation by splines with variables knots. *Springer*, 21:57–72, 1973.

[28] C. de Boor. Good approximation by splines with variables knots ii. *Springer*, 363:12–20, 2006.

[29] Carl de Boor. On calculating with b-splines* 1. *Journal of Approximation Theory*, 6:50–62, 1972.

[30] J. A. Dieza and L. Kondic. Computing three-dimensional thin film flows including contact lines. *Journal of Computational Physics*, 183:274–306, 2002.

[31] E.A. Dorfi and L.O C. Drury. Simple adaptive grids for 1-d initial value problems. *Computational Physics*, 69:175–195, 1987.

[32] B. D. Edmonstone, O. K. Matar, and R. V. Craster. Flow of surfactant-laden thin films down an inclined plane. *J. Engrg. Math.*, 50:141–156, 2004.

[33] B.D. Edmonstone, R.V. Craster, and O.K. Matar. Surfactant-induced fingering phenomena beyond the critical micelle concentration. *Fluid Mech.*, 564:105–138, 2006.

[34] B.D. Edmonstone, O.K. Matar, and R.V. Craster. Coating of an inclined plane in the presence of insoluble surfactant. *Journal of Colloid and Interface Science*, 287:261–272, 2005.

[35] B.D. Edmonstone, O.K. Matar, and R.V. Craster. Coating of an inclined plane in the presence of insoluble surfactant. *J. Colloid Interface Science*, 287:261–272, 2005.

[36] B.D. Edmonstone, O.K. Matar, and R.V. Craster. Surfactant-induced fingering phenomena in thin film flow down an inclined plane. *Physica D: Nonlinear Phenomena*, 209:62–79, 2005.

[37] B.D. Edmonstone, O.K. Matar, and R.V. Craster. Surfactant-induced fingering phenomena in thin film flow down an inclined plane. *Physica D*, 209:62–79, 2005.

[38] B.D. Edmonstone, O.K. Matar, and R.V. Craster. A note on the coating of an inclined plane in the presence of soluble surfactant. *Journal of Colloid and Interface Science*, 293:222–229, 2006.

[39] B.D. Edmonstone, O.K. Matar, and Craster R.V. Flow of surfactant-laden thin films down an inclined plane. *J. Engrg. Maths.*, 50:141–156, 2004.

[40] M.H. Eres, L.W. Schwartz, and R.V. Roy. Fingering phenomena for driven coating films. *Phys. Fluids*, 12:1278–1295, 2000.

[41] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to adaptive methods for differential equations. *Acta Numerica*, 4:105–158, 1995.

[42] J.M. Finn, G.L. Delzanno, and L. Chacón. Grid generation and adaptation by monge-kantorovich optimization in two and three dimensions. In Rao V. (Ed.) Garimella, editor, *Proceedings of the 17th International Meshing Roundtable*, 2008.

[43] R.M. Furzeland, J.G. Verwer, and P. A. Zegeling. A numerical study of three moving grid methods for one-dimensional partial differential equations which are based on the method of lines. *J. Comput. Phys.*, 89:349–388, 1990.

[44] R. Griffiths. The dynamics of lava ows. *Annu. Rev. Fluid Mech.*, 32:477–518, 2000.

[45] J. B. Grotberg. Respiratory fluid mechanics and transport processes. *Annual Review of Biomedical Engineering*, 3:421–457, 2001.

[46] J.B. Grotberg. Pulmonary flow and transport phenomena. *Annual Review of Fluid Mechanics*, 26:529–571, 1994.

[47] D. F. Hawken, J. J. Gottlieb, and J. S. Hansen. Review of some adaptive node-

movement techniques in finite-element and finite-difference solutions of partial differential equations. *Computational Physics*, 95:254–302, 1991.

[48] W. Huang. Variational mesh adaptation: Isotropy and equidistribution. *SAND82-8637, Sandia Labs, Livermore,Cal.*, 174:903–924, 2001.

[49] W. Huang, Y. Ren, and R. D. Russell. Moving mesh methods based on moving mesh partial differential equations. *Computational Physics*, 113:279–290, 1994.

[50] W. Huang and R. D. Russell. A moving collocation method for solving time dependent partial differential equations. *Applied Numerical Mathematics*, 20:101–116, 1996.

[51] W. Huang and R. D. Russell. Analysis of moving mesh partial differential equations with spatial smoothing. *Journal of Computational Physics*, 34:1106–1126, 1997.

[52] W. Huang and R. D. Russell. A high dimensional moving mesh strategy. *Applied Numerical Mathematics*, 26:63–76, 1998.

[53] W. Huang and R. D. Russell. The Adaptive Moving Mesh Methods. Springer, 2011.

[54] W. Huang and R. D. Russell. The Adaptive Moving Mesh Methods. Springer, 2011.

[55] O. E. Jensen and S. Naire. The spreading and stability of a surfactant-laden drop on a prewetted substrate. *J. Fluid Mech.*, 554:5–24, 2006.

[56] P. Keast and P. H. Muir. Algorithm 688: Epdcol: A more efficient pdecol code. *ACM Trans. Math. Softw.*, 17(2):153–166, 1991.

[57] L. Kondic. Instabilities in gravity driven flow of thin fliud films. *Siam Review*, 45(1):95–115, 2003.

[58] L. Kondic and J. Diez. Pattern formation in the flow of thin films down an incline: Constant flux configuration. *Journal of Computational Physics*, 13(11):3168–3184, 2001.

[59] L. Kondic and J. A. Diez. On nontrivial traveling waves in thin film flows including contact lines. *Physica D: Nonlinear Phenomena*, 209:135–144, 2005.

[60] R. LEVY and M. SHEARER. The motion of a thin liquid film driven by surfactant and gravity. *SIAM J. APPL. MATH.*, 66(5):1588–1609, 2006.

[61] R. Levy, M. Shearer, and T. P. Witelski. Gravity-driven thin liquid films with insoluble surfactant: smooth traveling waves. *European Journal of Applied Mathematics.*, 18:679–708, 2007.

[62] Y. Li, D. Jeong, and J. Kim. Adaptive mesh refinement for simulation of thin film flows. *Meccanica*, 49:239–252, 2013.

[63] G. Liao and D. Anderson. A new approach to grid generation. *Applicable Analysis: An International Journal*, 44:285–298, 1992.

[64] N. K. Madsen and R. F. Sincovec. Algorithm 540: Pdecol, general collocation

software for partial differential equations [d3]. *ACM Trans. Math. Softw.*, 5(3):326–351, 1979.

[65] P.K. Moore. Comparison of adaptive methods for one-dimensional parabolic systems. *Numer. math.*, 16:471–488, 1995.

[66] P.K. Moore. Interpolation error-based *a posteriori* error estimation for two-point boundary value problems and parabolic equations in one space dimension. *Numer. math.*, 90:149–177, 2001.

[67] K. W. Morton and D.F. Mayers. Numerical Solution of Partial Differential Equations. Cambridge University, 1994.

[68] T. G. Myers. "surface tension driven thin film flows," in the mechanics of thin film coatings, world scientific publ. *Co. Pte Ltd.*, 1996.

[69] T.G. Myers. Thin films with high surface tension. *SIAM Review*, 40(3):441–462, 1998.

[70] The Numerical Algorithms Group, Ltd., Wilkinson House, Jordan Hill Road, Oxford OX2 8DR, U.K. *NAG Fortran library, Mark 16A*.

[71] A. Oron, S. H. Davis, and S. G. Bankoff. Long-scale evolution of thin liquid films. *Reviews of Modern Physics*, 69:931–980, 1997.

[72] L. R. Petzold. A description of dassl: A differential/algebraic system solver. *SAND82-8637, Sandia Labs, Livermore,Cal.*, 1982.

[73] R. D. Russell, W. Huang, and Y. Ren. Moving mesh partial differential equations (mmpdes) based on the equidistribution principle. *Siam J. Numer. Anal*, 31(3):709–730, 1994.

[74] L. Shampine and M. Reichelt. The matlab ode suite. *SIAM Journal on Scientific Computing.*, 18:1–22, 1994.

[75] N.H. Shuaib, H. Power, and S. Hibberd. Bem solution of thin film flows on an inclined plane with a bottom outlet. *Eng. Anal. Bound. Elem.*, 33:388–398, 2009.

[76] G. D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*. Macmillan, third edition edition, 1985.

[77] P. Sun, R. D. Russell, and J. Xu. A new adaptive local mesh refinement algorithm and its application on fourth order thin film flow problem. *J. Computational Phys.*, 224:1021–1048, 2007.

[78] Tao Tang. Moving mesh methods for computational fluid dynamics. *Contemporary mathematics*, 383:141–173, 2005.

[79] E. Tuck and L. Schwartz. A numerical and asymptotic study of some third-order ordinary differential equations relevant to draining and coating flows. *SIAM Review*, 32(3):453–469, 1990.

[80] J. G. Verwer, J. G. Blom, R. M. Furzeland, and P. A. Zegeling. *A moving grid method for one-dimensional PDEs based on the method of lines*, chapter 12, pages 160–175. SIAM Philadelphia, 1989.

[81] E. Walsh. *Moving mesh methods for problems in meteorology.* PhD thesis, University of Bath, 2010.

[82] R. Wang, P. Keast, and P.H. Muir. Bacol: B-spline adaptive col-location software for 1-d parabolic pdes. *ACM Trans. Math. Softw.*, 30(4):454–470, 2004.

[83] M. R. E. Warner, R. V. Craster, and O. K. Matar. Fingering phenomena associated with insoluble surfactant spreading on thin liquid films. *Fluids Mech.*, 510:169–200, 2004.

[84] M. R. E. Warner, R. V. Craster, and O. K. Matar. Fingering phenomena created by a soluble surfactant deposition on a thin liquid film. *Phys. Fluids*, 16:2933–2951, 2004.

[85] A. M. Winslow. Numerical solution of the quasilinear poisson equation in a nonuniform triangle mesh. *Journal of Computational Physics*, 1:149172, 1966.

[86] Z. R. Zhang and T. Tang. An adaptive mesh redistribution algorithm for convection-dominated problems. *Commun. Pure Appl. Anal.*, page 341357, 2002.

[87] L. Zhornitskaya and A. L. Bertozzi. Positivity preserving numerical schemes for lubrication-type equations. *SIAM J. Numer. Anal.*, 37:523 – 555, 2000.

Appendix A

Selection of mesh density functions

A convenient choice of mesh density function is an essential basis for the success of an adaptive equidistribution method. Here, a interpolation error is presented for the adaptive equidistribution, which is dealt with in this report. The interpolation error has been selected for several reasons.

- It is easy to use.
- It contributes to saving time and effort of the computers memory.
- An estimate of truncation error can be formulated as a form of an estimate of interpolation error.
- It is one of the most reliable estimates

In general, we seek to understand how to choose the best mesh density function based on the error which is associated with the numerical solution of the partial differential equation used. For example, if approximating a function by a piecewise polynomial, the error is associated with whether a piecewise constant interpolation or a piecewise linear interpolation is used. Similarly, if finding the numerical solution of a partial differential equation, the error is associated with truncation error. Therefore, the mesh density function which will be described here is based on linear interpolation error. It

is claimed that the truncation error has the same form of linear interpolation error. So, the mesh density function used for interpolation error is also be valid for the truncation error for numerical solution of PDEs. In this case, the order approximation could be piecewise constant or piecewise linear and the error is measured in the L^2 norm. The basic idea is to choose a mesh density function based on minimising the error associated with the truncation error when finding a numerical solution of a PDE or interpolation error when approximating a function.

I have read and understood details and proofs given in Chapter 2 of [53]. I have not presented the details here but just listed the mesh density functions based on minimisation of error measure in the L_2 norm. So, the summary of the mesh density function:

Suppose that $\mathcal{J}_h$ is a mesh so that $\mathcal{J}_h$: $x_0 = a < x_1 < \ldots < x_N = b$, then

1. The error associated with a piecewise constant of a function u on a uniform mesh in L_2 norm is given by

$$N||u - \Pi_0 u||_{L^2(a,b)} \leq C \left[\int_a^b |u'|^2 dx \right]^{\frac{1}{2}}. \tag{A.1}$$

2. Error associated with piecewise constant of u on an optimal mesh density function in L_2 norm:

$$M(x) = \left(1 + \frac{1}{\alpha} |u'|^2 \right)^{\frac{1}{3}}, \quad \alpha = \left[\frac{1}{b-a} \int_a^b |u'|^{\frac{2}{3}} dx \right]^3, \tag{A.2}$$

$$\lim_{N \to \infty} N||u - \Pi_0 u||_{L^2(a,b)} \leq C \left[\int_a^b |u'|^{\frac{2}{3}} dx \right]^{\frac{3}{2}}. \tag{A.3}$$

3. The error associated with a piecewise constant approximation of u on an arc-

length mesh density function in L_2 norm is given by

$$M(x) = \left(1 + |u'|^2\right)^{\frac{1}{2}}, \tag{A.4}$$

$$\lim_{N\to\infty} N||u - \Pi_0 u||_{L^2(a,b)} \leq C \left[\int_a^b \frac{|u'|^2}{1+|u'|^2} dx\right]^{\frac{1}{2}} \left[\int_a^b \left(1+|u'|^2\right)^{\frac{1}{2}} dx\right]. \tag{A.5}$$

4. The error associated with a piecewise linear interpolation of u on a uniform mesh in L_2 norm, is given by

$$N^2||u - \Pi_1 u||_{L^2(a,b)} \leq C \left[\int_a^b |u''|^2 dx\right]^{\frac{1}{2}}. \tag{A.6}$$

5. The error associated with a piecewise linear interpolation of u on an optimal mesh density function in L_2 norm, is given by

$$M(x) = \left(1 + \frac{1}{\alpha}|u''|^2\right)^{\frac{1}{5}}, \quad \alpha = \left[\frac{1}{b-a}\int_a^b |u''|^{\frac{2}{5}} dx\right]^5, \tag{A.7}$$

$$\lim_{N\to\infty} N^2||u - \Pi_1 u||_{L^2(a,b)} \leq C \left[\int_a^b |u''|^{\frac{2}{5}} dx\right]^{\frac{5}{2}}. \tag{A.8}$$

6. The error associated with a piecewise linear interpolation of u on an arc-length mesh density function in L_2 norm:

$$M(x) = \left(1 + |u'|^2\right)^{\frac{1}{2}}, \tag{A.9}$$

$$\lim_{N\to\infty} N^2||u-\Pi_1 u||_{L^2(a,b)} \leq C\left[\int_a^b \frac{|u''|^2}{(1+|u'|^2)^2}dx\right]^{\frac{1}{2}}\left[\int_a^b \left(1+|u'|^2\right)^{\frac{1}{2}}dx\right]^2. \quad \text{(A.10)}$$

7. The error associated with a piecewise linear interpolation of u on a curvature mesh density function in L_2 norm:

$$M(x) = \left(1+|u''|^2\right)^{\frac{1}{4}}, \quad \text{(A.11)}$$

$$\lim_{N\to\infty} N^2||u-\Pi_1 u||_{L^2(a,b)} \leq C\left[\int_a^b \frac{|u''|^2}{1+|u''|^2}dx\right]^{\frac{1}{2}}\left[\int_a^b \left(1+|u''|^2\right)^{\frac{1}{4}}dx\right]^2. \quad \text{(A.12)}$$

In the above, $\Pi_k u$ indicates an interpolation operator of piecewise polynomials, k is a non-negative real number, called the degree of the interpolating piecewise polynomials on this mesh. For example, $k = 0$ corresponds to a piecewise constant interpolation and $k = 1$ corresponds to a piecewise linear interpolation.